EPISODE 773

[INTRODUCTION]

[00:00:00] JM: Zoox is a full stack self-driving car company. Zoox engineers work on everything a self-driving car company needs, from the physical car itself, to the algorithms running on the car, to the ride hailing system, which the company plans to use to drive around riders.

Since the company started in 2014, Zoox has grown to over 500 employees. Ethan Dreyfuss is a software infrastructure engineer at Zoox. He joins the show to discuss scaling and engineering team for self-driving. Machine learning was a big part of this conversation because there are so many different approaches that an engineering can take when it comes to machine learning for cars.

Can you take computer vision algorithms from academic papers off-the-shelf and apply them to cars? What about the computer vision APIs from the cloud providers? Can you use those for anything useful? What about physical world mapping companies, like Mapillary? There are so many options for build versus buy as a self-driving car company, and how do you do data labeling and data management? How do you manage the interactions across the stack? How do you manage the interactions between mechanical engineers, and user interface designers, and software engineers, and all of the other roles that go into building a full stack self-driving car company? We touched on some of these areas, but we barely scratched the surface of the self-driving car domain, and we hope to do that again in the near future.

[SPONSOR MESSAGE]

[00:01:43] JM: DigitalOcean is a reliable, easy to use cloud provider. I've used DigitalOcean for years whenever I want to get an application off the ground quickly, and I've always loved the focus on user experience, the great documentation and the simple user interface. More and more people are finding out about DigitalOcean and realizing that DigitalOcean is perfect for their application workloads.

This year, DigitalOcean is making that even easier with new node types. A $15 flexible droplet that can mix and match different configurations of CPU and RAM to get the perfect amount of resources for your application. There are also CPU optimized droplets, perfect for highly active frontend servers or CICD workloads, and running on the cloud can get expensive, which is why DigitalOcean makes it easy to choose the right size instance. The prices on standard instances have gone down too. You can check out all their new deals by going to do.co/sedaily, and as a bonus to our listeners, you will get $100 in credit to use over 60 days. That's a lot of money to experiment with. You can make a hundred dollars go pretty far on DigitalOcean. You can use the credit for hosting, or infrastructure, and that includes load balancers, object storage. DigitalOcean Spaces is a great new product that provides object storage, of course, computation.

Get your free $100 credit at do.co/sedaily, and thanks to DigitalOcean for being a sponsor. The cofounder of DigitalOcean, Moisey Uretsky, was one of the first people I interviewed, and his interview was really inspirational for me. So I've always thought of DigitalOcean as a pretty inspirational company. So thank you, DigitalOcean.

[INTERVIEW]

[00:03:50] JM: Ethan Dreyfuss, you are a software infrastructure manager at Zoox. Welcome to Software Engineering Daily.

[00:03:56] ED: Thank you very much, Jeff.

[00:03:57] JM: There may be people listening who are familiar with the self-driving car space but don't know much about Zoox. Describe what Zoox does at a high-level.

[00:04:08] ED: Absolutely. So I would say one thing that is unique about Zoox within the self-driving car space is that we're a pure play startup. So our goal is to, end-to-end, solve the autonomous vehicle problem. Really what that means is we're not just about the self-driving vehicles themselves. We're about delivering mobility people. So if you today would take an Uber or a Lyft to get to your location, really what we want to deliver is a wonderful experience to get from point A to point B.

Not only that, we want to deliver that safely. Really, I think, that’s a mission-driven piece of what we do is that safety piece. That's what brings a lot of us here. Something like 37,000 people died in the US in 2016 and that's something that we have an opportunity to go after here. We can bring that down by a factor of 90%. So that's what brings me to Zoox, and we have some really cool tech problems to solve besides.

[00:05:03] JM: How does the Zoox strategy for building self-driving differ from some of the other players like Waymo, or Tesla, or Cruise?

[00:05:12] ED: Absolutely. I think the biggest differentiator is that we are building our own battery electric vehicle from the ground up. We are also doing our own AI software to make it drive. We are building a service that will allow you to summon a vehicle. You're going to own, maintain and operate a fleet of vehicles and will do that in every single city that we’re operating in. Really, it's a pure play startup aspect of it that is I think quite unique to Zoox.

[00:05:41] JM: How does this difference in high-level strategy, how does it flow down to your personal work on software infrastructure?

[00:05:50] ED: That's what I think is one of the things that I really enjoy about Zoox as a whole and software infrastructure. We get to touch all of the pieces involved in making this happen. So this is something that is a difficult problem, and if anyone tries to tell you otherwise, then they're probably not tackling all of the pieces of this problem. So as an engineer, what fascinates me is the piece of this that is really tackling fundamentally difficult technology.

As an infrastructure manager, what that means is that I get to support the engineers and research engineers throughout Zoox in accomplishing this mission. That means that we as an infrastructure team get to tackle everything from building out machine learning platforms with Tensorflow, building out a supercomputing cluster, pieces on the hardware or software and product design side of things and really understanding how all of these fits together supporting these, right?

So you get a lot of these classic infrastructure problems around devops, around microservices. You get to build out a supercomputer. You get to understand how operations needs to work behind the hood to keep these services operating in a really, really high-level of reliability. So, to me, I think what's the most exciting about this is that I get to touch a lot of these pieces. I get to build out the team that is solving these fundamentally hard problems, and you get to push things a little bit further than you would see in in other environments. So we have to draw from multiple disciplines to solve the problems that we're trying to solve.

So you'll see I think this kind of separation in how a lot of big data is handled across different companies between the world of high-performance computing, for instance, and the sort of Hadoop style big data world and ecosystem that's built up around that. So we get to draw from both of those worlds to solve the problems that we're trying to solve. The scale that we're dealing with, especially around data.

So you look at these vehicles, these sensor that they're working with, it's really pretty incredible the volume of data that you're dealing with on a real-time basis. So we want to bring the best from these multiple disciplines together to be able to really solve these problems.

[00:07:59] JM: Most of our shows focus on pure software companies, and I want to get a better sense for what it means to work at a software car company. Tell me what a typical day looks like for you.

[00:08:13] ED: Absolutely. I'm not even sure that there is such a thing as a typical day, but just to give you a flavor of what some of the pieces are. So I'm walking in the door, I'm walking in a space that has a two-story opening to be able to fit a large C&C robot arm, and that's a prototyping device, and then I'm going to talk with the folks that are running that device and maybe there's some huge chunk of foam being carved out, and that's going to become a prototype part for the car or testing out the industrial design of the vehicle.

I think what you find is that that collaboration, that communication, is a really interesting piece of being in the space. It's certainly what's brought me to this space. I'm a software guy through and through. This is what I've done all my life, but coming back to the reason why I'm doing software is really at the end of the day to be able to impact the physical world to be able to make a difference in people's lives.

So I think you see that in a very direct kind of way when you're working with a hardware company and a software company all under one roof. Aside from that, we have some amazing devices that I just find fun. So I have a 3D printer at home and getting to work with kind of the next generation of some of these 3D printing technologies. I find that piece of it fascinating as well. Really, it's getting these questions answered.

So I think another piece of how that interaction works out in a day-to-day basis just to kind of give you an example here. So we moved into this space less than a year ago that we're in right now in Foster City, and one of the cool things was that was the first time we had all of those pieces of hardware and all of those pieces of software under one roof. That meant that week I sat down, I'm having dinner with someone that I hadn't talked to before, someone who joined the company more recently.

So I'm talking through the details of how lidar technology works. I'm learning about something that is not only the specifics of a sensor that we're using, but really how he's conceptualizing the design space for this class of sensors, differences around how you pulse light out, how you receive light back, and a lot of these actually makes sense to me, the software part of my brain. So I'm understanding that this is coding in the same way that you would do in coding of a datastream to be robust to noise if you had some kind of noisy channel that you are dealing with.

So it's really cool to get that kind of cross-collaboration. Then walking through the day-to-day of the software piece of it, I think one thing that's interesting about infrastructure software in particular is we become highly operational really quickly, especially as the company scales. So day-to-day, I'm coming in, I'm going to look at our metrics, our monitoring to be able to understand how are the infrastructure systems that are critical to what we're doing even at this stage before we've launched into the marketplace that are critical to the development work that our research engineers are doing, they're critical to operational work that focus on the people are doing. So I want to understand the status of those systems. I want to sort of get an update from folks on the devops side to find out if there're any things that I need to be concerned about there.

So I'll dive into metrics. I'll dive into monitoring. There's a real strategic piece to understanding what we want to be doing as an infrastructure team, and I think that's one of the interesting pieces of it too. Infrastructure at Zoox is sort of a little mini-startup within a startup, within Zoox a whole. So what I mean by that is that Zoox is going to be solving a very clear problem. We're not the kind of startup that's going to pivot one way or another.

We're solving autonomy. That's very clear to everyone. That's been the case since our inception, and nothing about that is changing. But infrastructure within Zoox, that's something where we really have to think what are going to be the high-impact problems and products that we can deliver as a team and how are people using those problems, or how are people using those products rather.

[00:12:09] JM: There are so many areas of software that you need to build. You've got, as you said, there's some monitoring and analytics stuff. There is of course the machine learning pipeline for gathering data and retraining your machine learning models. Can you just give me a break down of how you think about the different areas of software that Zoox divides into.

[00:12:37] ED: Yeah, absolutely. So I think at a top level you can kind of think about the pieces of the software that are running on the vehicle and then the pieces of the software that are running to make everything else happen. That's not quite the infrastructure versus other components of Zoox divide. I think really what that division comes down to is sort of resource constraints involved. What computers, what is the physics of what you're trying to do so.

On-vehicle, we have these tight compute constraints, and the things that you want to do with that on vehicle compute are solve motion planning and solve perception. So that's a very high-level division of the sort of on-vehicle component to what Zoox does. Now, what does it takes to actually support that from a software point of view? You need to understand what the communication is between those different components. So that's a team that is responsible for – We've called it a couple of different things, a software architecture, vehicle architecture, but currently we’re calling it kind of the core team. So that's another component of what we do that's on vehicle.

Then you kind of expand out into the world of what's outside of the vehicle, and that's where you get into infrastructure. So we’re everything to do with management of data off of the vehicle, management of both our cloud computing options and our on-vehicle simulation infrastructure. So by that I mean taking the same code that we would run on vehicle, running that really at scale off of the vehicle world.

Another piece of what we do that I think is maybe not obvious when people are looking at the problem as a whole in terms of how we approach vehicles, I think is that it's not just the code that you need to run on vehicle. It's all of the code you need to understand how to develop the code that you're running on vehicle.

So what do I mean by that? There's a really big effort that we have to do metrics to be able to really dive into anything that we see in terms of the behavior of the vehicles. Classify it in a very detailed way. Understand what we should be doing in terms of driving priorities. Do we want to be focusing on unprotected left terms, or are we already so good at unprotected left turns that we don't need to make any more improvements in that area?

I think at a high-level, some of these off-vehicle components, so we have the infrastructure team here. We've also got a team working on the simulation of the world off of the vehicle. We’ve got some teams working on sort of internal tooling as well, and I think that's a really interesting component of what we do. You really want to be able to visualize and dive deep into understanding scenarios that are going on on the vehicles. So it's I think that's a piece that's really interesting as well.

[00:15:29] JM: In my conversations with people who are working on self-driving, one distinguishing engineering feature of this space seems to be this feedback loop between gathering data from a car’s in meatspace experience, driving around, collecting data. Then the offline process of taking that vast quantity of data that a car has collected, putting it into a data platform, doing machine learning jobs to process that data, obviously that encompasses a huge space and there's “data engineering” that's going on on the car when it's just driving around.

Like you've mentioned that there is simulations that have to be run on the car that have latency constraints. So you can't necessarily make a round trip to the cloud.

But you have this large collection of problems that basically come from the fact that the car is gathering so much data. It has to be doing some processing on board for real-time updates and being able to maintain safety, being able to maintain the right navigation. But then you also just have this gigantic offline procedure, because you want to be able to do like high-bandwidth data engineering offline.

Can you give me a picture of the feedback loop between the car in meatspace driving around gathering data and that data getting to the cloud?

[00:17:08] ED: Yeah, I think that's a wonderful question. I think that really gets at the core of what we've found it takes to be successful in this space, and really it's investing in this system to go from meatspace to cloud. Actually I'll say, Jeff, that we're a hybrid cloud. So it's cloud data centers. It's also our own data centers as well that we need to get data to. I think the first piece of that is just the data volumes that you're dealing with.

So I think you've seen something really interesting happened over the last 5-10 years in terms of the trajectory of network bandwidth. So that goes back to kind of thinking of it from a physics and a meatspace constraints point of view. You have a certain amount of bandwidth that you can get off of these vehicles. So you start from, "Can I get the data that I need to off of the vehicle?" The answer sometimes is if I'm storing it in a format that I'm storing it – To give you an example, if I'm storing data coming off of our lidars in a format that doesn't make sense, I may actually blow out the bandwidth from that datastream past what any reasonable network link is going to be capable of.

So this is where you're talking maybe 100 gigabits. It is possible to aggregate multiple links, but if you go too far down that road, the rate at which you're generating data, it just doesn't make sense anymore. So I would say that's the place to start, is with these physical constraints on your system.

Once you sort of figured out the physical constraints, then I think you have another layer, which is collecting this data does no good if you can't make any sense of it. That's where you take the data in. You really kind of think through what are the queries that we're going to want to be able to run over this data. The pieces that you're going to find within that are everything from data mining. Being able to kind of dive in and find these rare scenarios.

So an example there would be something like yellow lights. So looking at yellow lights, that's going to be an event that is intrinsically less common than red or green lights, but you want a way to kind of dive into your dataset and pull out when you're training a machine learning model a uniform distribution from that. So that's going to inform the type of querying that you're wanting to be able to do on this dataset.

So, okay. What does that mean in terms of designing a system, a data architecture that will support that query? The answer is that it really kind of shapes the way that you store the data. It shapes when you run processing on the data. So what we're going to do once we get data off of the vehicle is run a fixed set of sort of preprocessing. You can think of it as a classic ETL type of step, except as part of that ETL we're running things like machine learning models and we're answering questions about what is really around in the world that the vehicle is operating in.

Then kind of the other piece of it is you want to get really good at aggregate understanding a fleet behavior. So this is where you're going to take a look at kind of everything having to do with behavioral metrics of the vehicles. How well are the vehicles able to make a particular type of maneuver? Again, it's an unprotected left turn, or it's being able to maybe nudge around a poorly parked car that is sticking out into the lane that you're currently in. These systems are made up of many different types of scenarios that they need to handle, and you really need to be able to do that aggregation and then rapidly answer questions about where are you prioritizing your development effort from an overall system point of view?

So I think one of the things that is really fascinating about this is you get to solve a lot of these problems and pull in the tools that you need to really make these other researchers, these other engineers, as productive as you possibly can.

[SPONSOR MESSAGE]

[00:21:10] JM: MongoDB is the most popular, non-relational database and it is very easy to use. Whether you work at a startup or a Fortune 100 company, chances are that some team or someone within your company is using MongoDB for work and personal projects. Now, with MongoDB Stitch, you can build secure and extend your MongoDB applications easily and reliably. MongoDB Stitch is a serverless platform from MongoDB. It allows you to build rich interactions with your database.

Use Stitch triggers to react to database changes and authentication events in real-time. Automatically sync data between documents held in MongoDB mobile or in the database backend. Use Stitch functions to run functions in the cloud. To try it out yourself today, experiment with $10 in free credit towards MongoDB's cloud products; MongoDB Atlas, the hosted MongoDB database service, and Stitch. You can get these $10 in free credits by going to mongodb.com/sedaily. Try out the MongoDB platform by going to mongodb.com/sedaily. Get $10 in free credit. It's the perfect amount to get going with that side project that you've been meaning to build.

You can try out serverless with MongoDB. Serverless is an emergent pattern. It's something that you want to get acquainted with if you're a developer, and getting that $10 in free credit to have a serverless platform right next to your Mongo database is really a great place to start. So go to mongodb.com/sedaily. Claim that credit, and thanks to MongoDB for being a sponsor of Software Engineering Daily. I love Mongo DB. I use it in most of my projects. It's just the database that I want to use.

Thanks to MongoDB.

[INTERVIEW CONTINUED]

[00:23:18] JM: If I wanted to build a self-driving car company today, I would want to take as much software off-the-shelf as possible, and I have no idea what is available today. What is out there? Are other good tools for simulations, or is there open source software for my robotic car control? What do you have to build and what do you have the option of buying in the self-driving car software world?

[00:23:47] ED: This is a great question. First off, I think I'm going to actually recommend that you not try to start a self-driving car company, not because we don't want the competition, but because I think it's actually a really difficult thing to start now. A few years ago, this was a very reasonable space to create a startup in. Today, I think everyone kind of understands the scale of the problem, and if you're starting from ground zero, you're not going to make it there in time as compared to the players that are in the space right now.

That said, there are pieces that exist in the market today that did not exist in the market three years ago, five years ago, and you want to be able to draw from as many of those pieces as you can. There are absolutely efforts both open source and commercial to solve pieces of this problem. Maybe I can break apart some of that, simulation for instance. There are for sure pieces that are built out either in the research community or in the open source community around doing simulation of vehicles. So that is something that you could find a version of off-the-shelf, but then dive into what that means and if that will fit your use case, many of those are actually built on a video game. So built on literally Grand Theft Auto, which is a really nice high-performance rendering engine. Does an amazing job of producing things that are, if not photorealistic, then at least very realistic outdoor street scenes.

You dive into that though, and the vehicle dynamics models are not going to be faithful to the real- world. So then you have to ask yourself questions about, "What does this mean? Is this something that can solve the problem that I actually am trying to use simulation to solve?" The answer may be, in some cases, yes. So it can do a really great job of taking your computer vision system and answering questions about the performance of that computer vision system. But if you're trying to test your motion planning system, for instance, that Grand Theft Auto based system may not really be a great answer there.

So I think one of the tricky things and one of the reasons why Zoox exists in the form that exists is we're trying to bring together these solutions, these experts from multiple disciplines, and really there isn't something off-the-shelf in most of these cases that will work at the level of quality that we need to be able to do what we need to do as a company.

So we’ve been able to bring in a lot of pieces, especially on the infrastructure side, that are these commercial products. So an example that is something that you can absolutely bring in in some kind of machine learning framework. You can bring in Café, you can bring in Tensorflow, and even once you have that, you then have to sort of think through, “Has that solved the problem for me? Maybe you need something on top of Tensorflow, something like CubeFlow that’s managing your workflow for machine learning. I think what you find is these problems get bigger and bigger, and the results is you’ve just got to respect the scale of the problem of building a self-driving vehicle. You’ve got to bring in a team that has the expertise that you need to be able to tackle this.

So one really interesting example I think is the mapping problem. So today, there are off-the-shelf solutions to mapping in the sense that there is really an ecosystem of startups that are trying to provide maps for self-driving vehicles. When you say a map, you mean something a little bit different than people might think of traditionally as a map. It looks like a street Atlas. It looks like Google Maps. That's a piece of what you need, but the other piece of what you need is an even denser 3D representation of the world.

So think something that looks more like a video game map that you might actually feed into Unreal Engine or something like that. So what do you do with that mapping software that map once you have it? It has to be really tightly tied into other components of your stack. So it has to be tied into how you’re running your perception system to understand where the vehicle is at any given time. So I think that's a really difficult position to be in to try to take a mapping solution off-the-shelf, bring that in and say, “This is going to solve self-driving for me, the mapping component of self-driving.”

The reason is the sensors that I'm going to be using, they’re different than the sensors that created that map. Their noise characteristics are going to be different, and they may not solve the problem in the way that I need it solved. Not only that, another interesting thing about mapping is you almost have to be able to solve that problem yourself. Otherwise, you're not going to be putting together all of the pieces that you need to do self-driving. If you're able to have vehicles operating at any scale at all, you can actually have them go down every single street within a region that you're operating in in a pretty short period of time, just a few days.

So, yes, you can bring in some of these pieces, but I think it would be a really difficult way to tackle this problem. I don't think you'd get them to mesh. I think where you'd kind of loose time as a startup is trying to do this integration effort and finding out that there are rough edges when you're plugging these pieces together.

[00:28:48] JM: So it's hard to take stuff off-the-shelf, at least as far as fully baked software is concerned. What about when it comes to research? There are lots of papers coming out in the world of computer vision, in the world of navigation. Can you take papers off-the-shelf and implement them, or do you feel like most of what you have to do on the research front, does that also come from in-house work?

[00:29:18] ED: So we have a world-class team doing our perception algorithms, our motion planning algorithms. What you'll find is a lot of them do have that academic background they published in the space. They've published papers that have won awards in this space. So we're absolutely drawing from that community. We actually have a couple of professors even, and we're drawing from the academic community, that academic literature, but we're taking it in a slightly different direction, which is we're trying to actually ship a commercial product.

What you'll find with a lot of the academic approaches is that they're solving a constrained version of a problem, and those constraints are really key to making that solution work. Maybe one great example of where the problem in an academic context versus an industrial context is a little bit different is this sort of ML research. So let's take the computer vision example. You kind of need a benchmark dataset. So you'll hear performance results on image net, for instance, reported. So everyone is working really, really hard to come up with the latest model and it's going to get a slight improvement on the image net dataset.

Absolutely, we will be reading those papers. We'll be taking a look at them. We have a great internal reading group where we discuss the latest work from academic literature, but maybe the right answer to solve this problem is not a more complex model architecture that you may or may not be able to fit in with all of the other components that you need to deploy to the vehicle. Maybe the right answer is actually dataset engineering. So you want to be able to double the size of your dataset and see what the performance implications of that are going to be on your model.

I think this is something that gets overlooked in the academic literature a lot is just how powerful it is to be able to reframe the problem in a way to be able to change what that dataset is, engineer the mixture of examples that you're feeding into your model. So as an infrastructure team, one of the things that we do is provide a lot of workflow support for that kind of research. So you're wanting to enable a research engineer within Zoox to, as rapidly as possible, explore an approach, decide if it's going to be effective, decide if initial results seem promising enough that we should continue to invest in it.

Then you go through this longer process of really optimizing that approach for performance. I mean, performance in two senses for the machine learning side of things. It's performance of the model with respect to precision and recall. So how accurate is that model? Then it's also a performance of the model with respect to code. So time, how quickly are we able to run the process of feeding in data and getting out an answer about where all of the vehicles in this image, for instance.

So, as you can imagine, we're not trying to solve this with an end-to-end neural network. There's not a single magical model that is going to take in images and radar and lidar and spit out a steering angle and a throttle or a target velocity, and that's just sort of the problem solved. We're structuring the problem.

Because of this structuring, we have a lot of different pieces that we're trying to fit on the vehicle. So sort of everything that we're shipping to the vehicle, it has to be able to justify itself from a runtime performance point of view, from how much GPU RAM is this thing consuming point of view, and then kind of pull that back into what is the impact on overall system performance? How is it improved our metrics as far as mission success?

[00:32:50] JM: What you're saying there, about not being able to build one big model to rule them all, this is not necessarily something that will be intuitive to people listening, certainly not necessarily intuitive to me. So you've got all these inputs from the world around you and the past examples that the car has seen or all of the Zoox cars have seen. You've got the weather conditions. You've got the current angle of the car. You've got the current surroundings. You've got the momentum, the acceleration. All these other things that are going on that you would take into account in the machine learning model that will end up – Or because these series of models that will end up emitting, “Here is the plan for the car to take.”

Can you help frame – I realized, it’s a very complicated subject.

[00:33:41] ED: How do you build an autonomous vehicle, right?

[00:33:43] JM: How do you build an autonomous vehicle? Okay. So in terms of coming to a conclusion of what is the plan for the car to take, like what are the rails that the car should follow? How can we frame this in terms that we can explain on a podcast? Is it like a sequence of models that you're synthesizing? But I guess it’s clearly not one big model. Can you help us clarify that?

[00:34:08] ED: Yeah, absolutely. So first off, I think your point earlier was that it’s not obvious that you can't do one big model, and I think that's true, right? So in theory, machine learning is becoming more and more powerful in theory. It could all be one big model. I mean, humans are driving vehicles in this way. They’re sort of one function almost. They’re thinking about all of these things simultaneously, and then you turn your steering wheel and your foot goes up and down on the throttle and the break.

So if machine learning were powerful enough today that we could just feed a bunch of inputs in and get out the output that we wanted, we would absolutely do that. NVIDIA, for example, has done some great work in a sort of research sense of showing that you can do this at small-scale. We’re not trying to do this at small-scale, but we’re trying to do this in a production way. What that means with the strength of the machine learning techniques that exist today is you have to provide structure to the problem.

So what is that structure look like? Well, you want to take an image in and you want to decide where all of the cars in that image. You want to decide where are all the bicyclists in that image? So this is the kind of structuring that I'm talking about, where you’re decomposing the overall problem into these smaller problems. I think I alluded to it slightly earlier, there is this high-level decomposition of the problem of self-driving that I think you'll find a lot of the players have, which is into the motion planning piece of it and the perception piece of it.

The perception piece of it is how do you take in all of the sensor data that you have around the vehicle and output something that's a sort of semantic representation of the world? So what that looks like is sort of a 3D notion of where the interesting things around the vehicle are. Interesting things may be lanes. Interesting things may be cars, pedestrians, pedicabs, street signs, traffic lights. So you want that level of semantic understanding.

Then sometimes you want to go even deeper than that. So it's great to have a system that will give you a perfect understanding of where all of the people around the vehicle are. That's not actually sufficient. You want to have some notion of where those people are headed in the future as well. That's another piece to kind of structure in this problem. Then once you have this semantic understanding of the world around the vehicle, you feed that into a motion planning system. What that motion planning system is going to do is say, "Given what's around the vehicle, given how that will evolve into the future, what do I do to achieve my mission goal?"

When we talk about a mission, it's sort of a simple type of thing. That mission is actually just get to a specific street address, right? Get to the Caltrain Station, whatever it is. So that's a high-level thing. It's a long way off in time and you have to translate that down into what does that mean over the next 10 seconds, the next 15 seconds, the next 30 seconds in terms of how I want to react? Should I be may be initiating a lane change now because I know I'm going to have to make a left turn coming up in a minute or two? So that's the sort of motion planning piece of it where you're trying to decide your high-level goals. How do they translate down into a lower level goal that will meet constraints like driving legally on the road that are hard constraints and soft constraints like getting comfortably and quickly to the destination that you're trying to get to?

So this is a very big space, but does that give you a little bit of that sense of how the problem decomposes?

[00:37:36] JM: Yes. I feel much more comfortable. Now, let's revisit the question of simulation, because this is – So one of these gigantic areas where you've got so many trade-offs you could explore, and one of those trade-offs is cost. I've heard the stimulation platforms can be really, really expensive to run. So you have to be somewhat judicious in terms of what you are stimulating? How aggressively you're simulating? What geos you're going to stimulate? Tell me more about your strategy for simulation and how it fits into the overall Zoox strategy?

[00:38:13] ED: Yeah. I mean, so one of the things that I think Zoox has thought about for a long time is the simulation problem. We've had a team dedicated to that effort since pretty close to our inception. The trade-offs that you're talking about, it really is that performance question, and the thing that it ties into that I think is maybe not obvious, this is really software release engineering. It's understanding what are the validation steps that we need to do to deploy a piece of software. So that might be a set of unit tests. That might be some integration tests. If I'm releasing a microservice, I might want to do a canary deploy and watch some health checks for a while before that release finishes. This is the same problem that we're trying to solve with simulation, but for self-driving vehicles. So it's a piece of the solution to this problem.

So we absolutely still have unit tests that run at one layer. Then above that, we have these integration tests that may have simulation as a key piece of how they're doing what they're doing. Simulation, you can have these differing levels of fidelity. So maybe you really only need a sort of 2D road network type notion of the world. That's going to give you a whole set of problems from, say, the motion planning algorithms, that you'll be able to surface those problems with that simulation, but that's a 2D simulation and it's cheaper than, for example, a full sort of 3D model of the world. But if you don't have that 3D model of the world, how are you possibly going to understand how your perception and vision system is going to handle occluded pedestrians suddenly walking out from behind a parked vehicle for instance?

So that's where you need that sort of next level of fidelity and simulation, and those get expensive, and that ties very closely to what we do on the infrastructure team, is we've got to be able to scale out some of these techniques. We don't want other teams within Zoox to be limited by the compute side of things. We want to give them the most they can possibly have so that they can use some of these expensive techniques, because 3D simulation is a really useful technique to be using, and we need to be able to do it at scale. We need to have a portfolio of scenarios that we're simulating that are going to be representative of the – We call the operational design domain, the region either geographically or in terms of sort of behaviors that we know the vehicle is capable of handling. That really covers a very broad spectrum of different scenarios.

So I think you've hit on the trade-off here and the hard part of it. We're willing to throw the most compute that kind of exists within the world. We're willing to throw supercomputing levels of compute at this problem, but we can't throw more compute than that at it. So how do you be judicious about the usage of that compute and how do you make sure you're getting the most out of every usage of that compute?

[SPONSOR MESSAGE]

[00:41:12] JM: We are all looking for a dream job, and thanks to the internet it's gotten easier to get match up with an ideal job. Vettery is an online hiring marketplace that connects highly qualified job seekers with inspiring companies. Once you have been vetted and accepted to Vettery, companies reach out directly to you because they know you are a high quality candidate. The Vettery matching algorithm shows off your profile to hiring managers looking for someone with your skills, your experience and your preferences, and because you've been vetted and you're a highly qualified candidate, you should be able to find something that suits your preferences.

To check out Vettery and apply, go to vettery.com/sedaily for more information. Vettery is completely free for jobseekers. There're 4,000 growing companies, from startups to large corporations, that have partnered with Vettery and will have a direct connection to access your profile. There are full-time jobs, contract roles, remote job listings with a variety of technical roles in all industries and you can sign up on vettery.com/sedaily and get a $500 bonus if you accept a job through Vettery.

Get started on your new career path today. Get connected to a network of 4,000 companies and get vetted and accepted to Vettery by going to vettery.com/sedaily.

Thank you to Vettery for being a new sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:42:57] JM: Is your perspective on simulation, is it more that this is a tool that’s useful for like the testing before release process, or do you also see it as useful for training data? Because I remember reading this article, I think was in the Atlantic about Waymo and about how simulation is so core to how they even generate training data, like they’ll kind of map the physical world and then they will run simulations of the – This is sort of like the Grand Theft Auto example, just run simulations of the way that the car would behave in that simulated environment, and that's in order to gain more training data, because it's like, “Okay. Now we’ve got training needed in the real-world and we've got additional training in the simulation.” That's a little bit different than the kind of integration test or unit test approach to simulation.

[00:43:51] ED: Yeah. So I made the analogy to a release process, because I think in many respects, it is the same kind of problem, but actually the stack keeps going up from there. So you have things like hardware in the loop test. So that's the release process piece of it. There's also what you're talking about, which is a development process, and you absolutely have to optimize for that development process.

So that will look like iteratively an engineer on the planner team or the vision team is making a change. They want to understand the impact of that change, and they want to do it without having to consume a lot of time, right? So as soon as you want to run a test on a vehicle, that's going to be a much more costly process, and you want that development workflow to be as short as possible.

So that the web development world now is amazing, because I type a change and it’s live on my hot reloading view of the website that I'm editing. You want to get as close as possible to that kind of iterative experience for someone who is developing a planner algorithm or a vision algorithm. Then there's this extra piece of it, which is the piece you're talking about, which is simulation can actually sort of push forward what we’re capable of doing for computer vision algorithms, for instance, because it gives us this wonderful, perfect label ground truth.

Again, I was talking earlier about how you decompose this problem into multiple pieces. You decompose it into, for example, finding all of the vehicles in an image. To do that, machine learning, the lifeblood of a machine learning algorithm is data, and there are some approaches that can handle unlabeled data, but by and large, the approaches that work the best are going to consume huge amounts of labeled data. What this means is that I have an image that I'm able to feed in where I know the ground truth, I know where all of the cars are in that image.

To get that, it is an expensive process. You have to have a human tell the system what are the boxes. So select a corner. Select another corner. That kind of labeling is important. If it's a video game world though, I know the extent of that vehicle. I've rendered that vehicle and I can render this sort of bounding box corners of that vehicle and get this wonderfully perfect ground truth.

We've actually published a couple papers in this area and we've shown that you can extend these systems. You can go beyond just using the raw data labeling, the raw data that's coming in from the real- world. You can use data that is produced in a simulated world and improve your results on the real- world, and it's sort of not obvious that that would work for sure. That depends on the fidelity of your simulation. What we've shown is that we've been able to get our simulation to the level of fidelity you need for that type of approach to be helpful as well.

[00:46:32] JM: Okay. Now that we've framed the self-driving problem to I think a reasonable extent, we could talk a little bit more about data infrastructure.

[00:46:42] ED: Data infrastructure. I love it. That's my bread-and-butter.

[00:46:46] JM: Exactly. It's much closer to my bread-and-butter too. I'm so out of my comfort zone talking about self-driving car software, because I've done so few shows on it. I'm really glad we're changing that here.

On the date infrastructure side, like we've done shows with Netflix and Uber and some other companies that feel a little bit more familiar to what I understand, at least. But even here, you're in a unique position, because you can take this really long term perspective. You're not like having day-to-day firefighting operations where like people – At Uber, for example, I mean that's a production system that needs to be ready for people right now. We need this data platform yesterday. We've got services that are live. We've got things we need to build on top of the data platforms. We've got logging. We've got monitoring, and that stuff is also important for you.

But you can take a bit of a longer perspective, because you know there's a lot of money in the bank and it's a service that's not really released to consumers yet. So it's kind of still in the lab and you can take this longer-term approach. Tell me how you frame your approach to data infrastructure and get into some of the tooling that you are using.

[00:48:00] ED: Yeah! So one of the questions that I had to answer back three years ago when we were first creating an infrastructure team at Zoox is really what does that team do. What problems can we solve to make a difference to this company?

I think one of the first places the we've found that we could really dive in and make a difference was optimizing for the developer, experience the researcher experience, making those workflows as fast as they possibly can be. I think having this almost sort of user experience-focused viewpoint for a product that is internally facing rather than externally facing, that's been kind of the key to doing what we do.

Now, there are pieces to it that are operational already, and I think what you find is on the infrastructure side we got to that operational problem domain much sooner than maybe some other parts of Zoox, and the reason is as soon as we were able to successfully solve problems for other engineers within Zoox, those become a critical part of those engineers' workflow. Meaning that if that service is down, I've now impacted the productivity of a significant portion of the company.

So I've built out an SRE team here within infrastructure at Zoox, and that's been key. That reliability component has been key to what we're doing. That said, I also get the opportunity to kind of step back and say where are things going from a more traditional infrastructure point of view. What are some long-term trends either in hardware or software that drive what we should be doing from a solutions point of view?

So we've really wanted to tie into some of the best practices around devops. Being able to pull from microservices and really think hard about a rapid release and a robust software engineering practice and bring that to what we do within an infrastructure team, because as soon as something that we've done has been successful within Zoox, that is critical infrastructure. Yes, it's not serving a customer, but our customers are our other engineers today, and their productivity is key.

[00:50:13] JM: Can you give me descriptions of some of the tools that you're using? So maybe cloud providers, open source tools. I don't know, Spark, or Flink, or Tensorflow. Give me some sense the lay of the land.

[00:50:26] ED: Yeah. So I think one of the pieces that we tried to solve really early is an infrastructure team is data pipelines. So you can think of this sort of as an ETL type pipeline where data has come in off of the vehicle and you want to be able to run some type of data processing on it. So the way we've chosen the model, that it's not the only way, but the way we've chosen a model that is as a data pipeline. So there you can think of a system like Airflow for instances is one example of a workflow management system. Uzi is another example.

So that is sort of the high-level representation of the processing that we're trying to do on this data coming in. Then the thing that you have to do is schedule that compute job in some way. So there we did a lot of evaluation around the best way of scheduling a group of machines, and this is a problem that is absolutely one where we want to draw from the industry from pieces that are out there. So we were looking at things like Kubernetes, Mesos.

We ended up using a component called Slurm that's actually coming out of the HPC world. So this would be something maybe a little bit more akin to a resource manager. So the different ways of slicing and dicing this problem was Slurm is all about is taking a job with some specification for the resources required for that job and allocating, packing those jobs on to a group of machines with different capabilities.
So we've kind of built out some tooling, some integration on top of that to make it really easy to use within Zoox.

One of the things that we do is we do all of our development on GitHub. So we built out some integrations between Slurm and our internal GitHub to make it really easy to sort of kickoff a job that is based on a specific version of the software that we're using. But at the end of the day, we need that component that is doing the resource management, and I think one of the questions that I've gotten is why use that, which is a little bit less known coming out of the HPC world than something like Kubernetes, which especially the last couple of years has gained a tremendous amount of momentum? I think there're some detailed answers to that that become interesting for the types of problems that we're tackling.

So we're really focused on using GPUs as computer accelerators. This is one of those long-term trends where we do get to take that long view a little bit and say, "Compute is becoming available in compute accelerators in a way that hasn't been the case in the past necessarily. I've been able to do an awful lot with x86 CPUs, and that architecture has been a stable platform. I haven't had to worry about adding in something that is a very different computer model, a very different piece of hardware. Well, now I do, and that means I'm having to think of the computers, the nodes that I'm running on the servers that I'm running on as themselves mini-networks. Because the HPC world has been doing this for a long time, we were driven a little bit to try to draw from that world.

So what I mean by a mini-network is GPUs themselves are all communicating with the host CPU and they're doing it over the PCIe bus, or in some cases there's an NVIDIA proprietary standard, NVLink that is also a communication path that's involved. So you have to be able to be aware of the impact of scheduling a job that might run on multiple CPUs on GPUs that have a longer communication path from a PCIe topology point of view.

That's something that was baked into this software, because Slurm has been used for a very long time in this HPC world. That said, nothing bad to say about Kubernetes. We're using it internally and it's the momentum behind that community is incredibly strong right now, and I think it may actually start to develop some of these capabilities in the long run. I think it is already very possible to schedule GPUs with Kubernetes for instance. It's just some of these more detailed cases that are a little bit difficult to do there.

[00:54:10] JM: So there's so many questions I would like to ask you about data infrastructure. Maybe we can do another show in the future about this, but I would be remiss if I didn't ask you some questions that require a baseless speculation.

[00:54:22] ED: Oh, of course.

[00:54:23] JM: So what will the self-driving market look like in five years?

[00:54:27] ED: Oh, in five years. So I think what you'll find is that these are a part of everyday life in five years in a way that is sort of hard to appreciate in the same way that understanding what having a cell phone part of your everyday life would have meant five years before they became common, or maybe even a better analogy, understanding what a smartphone would be like five years before the iPhone.

I actually don't think we're were sort of five years before the iPhone in the sense that the first products are not five years out, but I think we're far from going to be passed from the sort of ubiquitous stage where people completely forget about these things. So I think five years from now, this will be a part of everyday life for many people, maybe most people even, but they won't be everywhere. There will be limits to these systems still.

So we talked a lot about this notion of an operational design domain. This is this notion that there may be pieces of the overall driving problem that just don't make sense to go after from a commercial, from a business point of view. So an example here is I've seen lots of questions around, "Well, I'm never going to lead a self-driving car drive me, because it's not going to be able to handle snow like I could on a mountain road in rural Colorado," and maybe that's right for the systems that are going to be out in the next couple of years. Probably if we were to focus in on that, we could solve that problem, but is that the right problem to be going after commercially? I think the answer is no. The volumes of travel, the volumes of trips there, they're less common.

So I think looking out five years, what you'll see is it's a part of everyday life. You see it many places, but it's not everywhere, and these systems are getting better, really, really rapidly. It's still a very active area of development. I think one thing is very cool about that for people that are kind of coming into this space now is there is a career here and it's not like we're a couple years out from being done. It's sort of a couple years out from the start of this and there'll be some very fierce competition in this space too.

I think you'll see a handful of players. There'll be a real shakeout where folks that haven't been able to achieve the results that the top couple players have been able to achieve may not be able to sort of continue to operate. There'll be some consolidation that happens in the industry. Then folks that are able to really deliver the technology in a way that you see from a place like Zoox will be continuing to iterate on what we're doing tackling more and more problems that are these pieces that are maybe not in that MVP product, but expanding the operational design domain that we're capable of operating within.

[00:57:10] JM: There were a number of years where the only people you would hear about who were working on self-driving cars were like somebody that worked in a Stanford lab with Sebastian Thrun, or I don't know if that guy was at MIT, wherever he's from.

[00:57:25] ED: Yeah, he was at Stanford.

[00:57:27] JM: Stanford, right, or like somebody who is it an MIT robotics lab, or somebody that was like a higher up at Google who happened to get in at Waymo, or like these superstar engineers. I think there's probably the lesser understood side of self-driving, where you can be a pure software engineer with a background at pure software companies. There is a need for you in the self-driving car industry today.

So if there's engineers who are listening to this and they're listening to this and they're like, "That's not me. I can't do self-driving engineering." Give any advice for people on like how to get into the self-driving car industry. Are there any hurdles? How should a terrestrial engineer think about this?

[00:58:16] ED: That's a great question. So I think what is unappreciated sometimes is that even the teams that are comprised of a lot of people from that background, a big chunk of what those people are doing looks more like traditional software engineering.

So you are building out software systems that have to be reliable, that have to be scalable. Other engineers need to be able to read and understand your code. It has to deal with data in ways that are maybe feel the same as dealing with data that's a click stream for an advertiser or something like that.

So you absolutely need this component of what we're doing in terms of how to get into the industry. I talked to a lot of people that haven't thought deeply about the self-driving space when they're applying to us, when they're first talking to me, for instance. I think it is fine to come in with that background. So I would say give it a try. There may be more opportunities sooner than you think.

Then what I see is as sort of an even better nice to have is someone that's sort of thought about this space, read some of the news articles that are out there, understand trade-offs in different approaches. So really, I would say the first place to start more than building up a specific skillset – I mean, sure, there are technologies and tools I would like people to be familiar with coming in, having people that understand data engineering, but I think the problems space is so big. The thing to remember is come in with an interest in the space. Understand the impact of what we're trying to do with self-driving vehicles, and then see where your skills that you already have can be applied to that problem.

So if you can do that extra step of thinking a little bit, then I think you'll find a place for the skills that you have. If you have a really strong devops background, for instance, I absolutely am looking for people with that skillset. If you have a really strong background in maybe formally verified software, I mean, that's one's a little bit different, but it's a world where we need people with that skillset.

Do you have a strong background in microservices? We absolutely need people with that microservices background being able to deploy the services that are operating the fleet of vehicles that we have. So I would say the question is almost not how do you upscale to be able to get into the self-driving industry, but where can you apply the skills that you have? How do you think enough about it to kind of understand how those skills will fit into what you're doing.

[01:00:44] JM: Ethan, thank you for coming on Software Engineering Daily. It's been really fun talking to you.

[01:00:47] ED: Jeff, thank you so much. It's been a real pleasure.

[END OF INTERVIEW]

[01:00:53] JM: This podcast is brought to you by wix.com. Build your website quickly with Wix. Wix code unites design features with advanced code capabilities, so you can build data-driven websites and professional web apps very quickly. You can store and manage unlimited data, you can create hundreds of dynamic pages, you can add repeating layouts, make custom forms, call external APIs and take full control of your sites functionality using Wix Code APIs and your own JavaScript. You don't need HTML or CSS.

With Wix codes, built-in database and IDE, you've got one click deployment that instantly updates all the content on your site and everything is SEO friendly. What about security and hosting and maintenance? Wix has you covered, so you can spend more time focusing on yourself and your clients.

If you're not a developer, it's not a problem. There's plenty that you can do without writing a lot of code, although of course if you are a developer, then you can do much more. You can explore all the resources on the Wix Code's site to learn more about web development wherever you are in your developer career. You can discover video tutorials, articles, code snippets, API references and a lively forum where you can get advanced tips from Wix Code experts.

Check it out for yourself at wicks.com/sed. That's wix.com/sed. You can get 10% off your premium plan while developing a website quickly for the web. To get that 10% off the premium plan and support Software Engineering Daily, go to wix.com/sed and see what you can do with Wix Code today.

[END]