EPISODE 1084

[INTRODUCTION]

[0:00:00.3] JM: Uber needs to visualize data on a range of different surfaces. A smartphone user sees cars moving around on a map as they wait for their ride to arrive. Data scientists and operations researchers within Uber study the renderings of traffic moving throughout a city. Data visualization is core to Uber and the company has developed a stack of technologies around visualization in order to build appealing, highly functional applications.

Deck.gl is a library for high-performance visualizations of large data sets. Luma.gl is a set of components that targets high performance rendering. These and other tools makeup Vis.gl, the data visualization technology that powers Uber. Uber's visualization team included Ib Green, who left Uber to co-found unfolded.ai, a company that builds geospatial analytics products.

Ib joins the show to discuss his work on visualization products and libraries at Uber, as well as the process of taking that work and founding unfolded.ai. Full disclosure, I am an investor in unfolded.ai.

If you want to reach 30,000 unique engineers every day, consider sponsoring Software Engineering Daily. Whether you are hiring engineers or selling a product to engineers, Software Engineering Daily is a great place to reach talented engineers. You can send me an e-mail, jeff@softwareengineeringdaily if you are curious about sponsoring the podcast.

We're also looking for writers and a videographer. If you're great with video, or you're a great writer and you want to write about software, then send me an e-mail, jeff@softwareengineeringdaily.com.

[SPONSOR MESSAGE]

[0:01:53.6] JM: SAP is a company that touches 23 trillion dollars of consumer purchases around the world. You've probably heard of SAP, but you may not know about SAP’s open source investments, such as SAP Data Intelligence.

SAP Data Intelligence connects and transforms data to extract value from the distributed data landscape and embraces the best of open source technology. SAP Data Intelligence brings together data orchestration, metadata management and powerful data pipelines with advanced machine learning, enabling close collaboration between data scientists and the rest of your infrastructure teams.

SAP Data Intelligence runs on Kubernetes. SAP's contribution to Kubernetes includes the Gartner Project, that helps to operate, monitor, manage and keep Kubernetes clusters alive and up-to-date.

To learn more about SAP Data Intelligence, you can visit sap.com/sedaily. That's sap.com/sedaily.

[INTERVIEW]

[0:03:03.9] JM: Ib Green, welcome to the show.

[0:03:05.5] IG: Thank you.

[0:03:06.6] JM: You were at Uber for four years on the visualization team. Describe what you worked on at Uber on the visualization team.

[0:03:14.8] IG: Yes. At Uber, we had very interesting challenge. We had a very big data, obviously. I tend to think of Uber as a planetary scale transportation system that generates enormous amount of data every second. An aspect of that data is at all that data has a geospatial information associated with it, everything; all the trips takes place somewhere and they start somewhere and they end somewhere. Of course, when you operate at that scale as Uber does, it's extremely important to develop efficiencies. We want a very efficient operation.

To be able to derive insights from the data and understand how various actions that you take actually affect the efficiency would then be extremely valuable. The other thing that we also realized in the visualization team was that with WebGL technologies today, you can access the

GPU from the browser and visualize very large data very seamlessly and easily. What we set out to do in the visualization team was to build a set of the technologies, a set of frameworks to visualize big data directly in the browser, especially geospatial big data.

[0:04:26.0] JM: Could you just double down on why visualization is so important at Uber? You have visualizations, not just in the Uber app itself, like the visualization of a car moving towards you, but you have lots of back-end visualizations. Tell me more about the span of different visualizations across Ubor.

[0:04:45.5] IG: Yes. Visualization has been a very powerful concept for out Uber. It obviously has a very wide range. There's a lot of a traditional visualization, which is traditional charting and you have the various types of graphs showing doing of the business performance or other types of metrics, especially the geospatial visualizations, where we're showing data on top of maps. There is many, many instances. There's probably four dozen major applications built inside Uber on top of these frameworks and there's everything from seeing where costs are in specific region to being able to narrow in for a customer support first and to narrow in a specific trrip and see exactly when different things happen, where the car was, when to call, when it was dispatched and when the pickup happened, where the drop-off happened and other things.

There is machine learning pipelines where various types of improvements are being made and you want to go in and see where and understand what's happening. As the trip is being optimized, obviously Uber has developed autonomous vehicles, so there's some very rich amount of visualization happening there. There's maps, very good understanding of maps also requires visualizing data on top of the maps. Uber elevate has helicopters and drones and needs to look at their channels and other types of compliance. It goes on. There are many more examples.

[0:06:07.7] JM: As far as displaying large quantities of data and displaying data in a geospatial arrangement, I assume there are some prototypical visualization problems. You've got vast quantities of data, you've got data that you need to render quickly, you've got streaming data, you've got lots of different application surfaces. Can you tell me the prototypical visualization problems that you encountered at Uber?

[0:06:38.1] IG: Yes. There are obviously many and it depends obviously on the situation that you're in. We tried to create a fairly general tool, so we tried to build systems that would work on general tables and be able to display very large tables without a lot of additional processing. Obviously, there's everything from you need a lot of work on the data side obviously to create data pipelines and you need to process the data in it to ensure it has integrity as it reaches the point where you want to visualize it and you need to have the good system to stream the data into the browser and get it ready for visualization.

If you look more on the front-end side where we did a lot of the work, utilizing the GPU, if you look at traditional 3D frameworks, they're more to designed for games and seeing growth type designs and they tend to allow very sophisticated manipulation of the different elements in the scene, but they also require a traversal with individual draw calls. That really only scales well to maybe a few thousand elements on the scene. The technology that we use for efficient visualization is something called instance rendering. It was really developed for games where you wanted perhaps to have a very large set of soldiers, or lots of similar types of vegetation, or straws of grass or something and you render basically one primitive again and again and again, but just with a slightly different parameters, or different position, a little bit of a different filter angle.

We basically upload entire tables, large million row tables directly into the GPU and then we treat them as we use this instance rendering techniques to get very, very high levels of performance.

[0:08:19.4] JM: Giving the browser access to the GPU, that's a newer development that when the browsers had access to GPU?

[0:08:30.0] IG: I think roughly around 2011 maybe, the first versions of WebGL started to become available. There were some early frameworks that were started to utilize that really already from the beginning. For instance, 3JS, which actually everyone now knows maybe as the leading JavaScript WebGL framework, had existed even before WebGL and had different types of renderer, more of a canvas renderer and then added a WebGL renders. That's interesting. There's also other types of geospatial frameworks like CCM JS for instance, that have been around for a long time.

[0:09:08.1] JM: Can you tell me more about the interface between the GPU and the browser?
[0:09:13.8] IG: The really exciting thing I think with the ability to access the GPU is that normally when you're in a browser, you're in a virtual environment. You typically would expect to have a certain amount of performance penalty compared to working with native application. Everything is either interpreted or just-in-time compiled, but they're still and not the end. It's arguable. When you work with a GPU, you basically upload data buffers and then you provide various types of [inaudible 0:09:42.5] programs and description of how to process those.

Once you start the GPU, there are no inefficiencies anymore. You're basically, you can't get closer to the metal than that. It's very powerful to be able to work directly with the GPU. The small limitation that you have is that you have to work with the WebGL API, which does not give access to all the features of the latest reviews. That's something of a limitation, but that's also being addressed with the web GPU and other technologies that are now emerging.

[0:10:13.2] JM: Could you give me an overview of the state of geospatial visualization tools, because you've been working on this space since 2014? I know a lot has changed since 2014 and also, when you actually were developing tools at Uber, there was some prior art. Can you give me a brief timeline of geospatial visualization tools from when you've been in the business?

[0:10:37.5] IG: Yes. Well obviously, there are or companies that have been around for a very long time. Ministry has been doing a geospatial tooling for maybe almost 50 years. The timeline is quite long. I think when it comes to doing these things in the browser and taking advantage of WebGL, I think a big and obviously CCM JS, which has been more targeting full 3D globe visualizations with maybe aerospace that type of originating in those type of used cases.

Then of course, Mapbox, who did an incredible job on rendering map tiles or base maps with very high performance with Mapbox or GL. To me, those are maybe the most important comparisons. There are obviously other companies that provide a more full services with back-ends and other things to process your spatial data.

[0:11:32.8] JM: Yeah. You mentioned Esri, they make the ArcGIS system. We actually just did a show with that. Can you tell me about what role ArcGIS plays in the geospatial ecosystem?

[0:11:47.3] IG: Well, I mean, I think it plays a huge role, right? Esri is an incredibly established company with an enormous user base, hundreds of thousands of customers. They have a very big trade shows with and they also there – maybe 70,000, 80,000 attendees actually coming in visiting their shows. I think their developer conferences also tend to be very, very well visited.

They built a huge ecosystem around this. Obviously, they've been around for a very long time and they cover a very wide range of use cases. It's really hard for me to pinpoint something specific around Esri, because I think they have some offering and some product in almost every geospatial facets.

[0:12:28.8] JM: I guess, we can get to Uber. You started at Uber in 2015. Around that time, React was becoming quite popular. I think no discussion about the front-end would be complete without a mention of React. How important was React in developing high-quality visualization tools around that time?

[0:12:49.9] IG: Yes. React was a very influential at Uber. The front-end community at Uber really came together around using React and we were able to create components and frameworks that could be shared across all applications, because of these. Because of that, it was it was very influential for the design of the visualization frameworks and technologies and especially, the aspect of React that it is based on a functional programming paradigm, which are very powerful for user interfaces, but it's not well supported by many traditional, more imperative 3D frameworks. Being able to do GPU programming in a way that is compatible with functional programming paradigms then became one of the design criteria for our frameworks.

[0:13:31.0] JM: What is the interaction between React and WebGL? Can you talk more about WebGL?

[0:13:37.9] IG: Yes, absolutely. The way we think about WebGL obviously is that we have these very big – once you hand over to WebGL or to the GPU rather, you need to set up big tables of data that are formatted in on the GPU, uploaded to the GPU, the buffers of data that are formatted in just exactly the right way that the various layers shaders expect. The work of

preparing these tables and uploading them is then quite costly. A tables could have millions of rows, so this can be many megabytes of data.

You don't want to update these tables any more frequently than you absolutely need to. This is a big challenge in the functional programming style, where you have to make sure that as you declaratively update the layers, now the visualization, the framework needs to very intelligently make decisions about when the data needs to be uploaded to the GPU, or modified, or recalculate and updated on the GPU.

[0:14:33.5] JM: Yeah, tell me more about how WebGL plays a role in visualization tools.

[0:14:38.4] IG: Well, the basic concept is to be able to bring in very big data without having to do a lot of processing on it. To some extent, the WebGL, if you think about for those who know how a shader works, the shader basically have different attributes and could have big positions, colors and other things, the data that you feed into the shader. For us when we think about tables, the traditional tables, which you normally think about in row-based form, we think about those in columnar form. You have a table consisting of columns.

Mapping those columns onto the attributes, it's basically we think of the GPU as having a columnar table in GPU memory and we think of the data that's coming in as being a columnar table in CPU memory, or on disk. The framework does the mapping between these two and then we basically write shader code to render this very efficiently.

Obviously, if you work with more on the CPU side, you can work with technologies such as SVG, for instance. SVG has a lot of styling that you can apply. Once you get to a 1,000 elements, or 10,000 elements of SVG, your performance becomes quite challenged. We need to then – the goal is then to provide a subset of that styling, but render things on the GPU.

There's two times to do styling on a GPU. You can do it through uniforms, which means you can change them very quickly, but the properties, the changes that you make will then apply to all the different elements in your table. If we have a scatterplot layer re-rendering one point per layer and you can set the uniform and you can change the radius scale of all the points for

instance, or the opacity of all the points very, very efficiently. You can animate those changes very efficiently.

If you want to make the change to each individual element, then you can. It's a terminology called, often referred to as data-driven styling, so that the contents of each row essentially in the table would affect the styling of the elements corresponding to that row, to that point. Then we have to recalculate the attributes in the GPU table. Those are some things to think about there.
[SPONSOR MESSAGE]

[0:16:44.7] JM: Over the last few months, I've started hearing about Retool. Every business needs internal tools, but if we're being honest, I don't know of many engineers who really enjoy building internal tools. It can be hard to get engineering resources to build back-office applications. It's definitely hard to get engineers excited about maintaining those back-office applications.

Companies like DoorDash and Brex and Amazon use Retool to build custom internal tools faster. The idea is that internal tools mostly look the same. They're made out of tables and dropdowns and buttons and text inputs. Retool gives you a drag-and-drop interface, so engineers can build these internal UIs in hours, not days. They can spend more time building features that customers will see.

Retool connects to any database and API. For example, if you want to pull in data from PostgreSQL, you just write a SQL query. You drag a table onto the canvas. If you want to try out Retool, you can go to retool.com/sedaily. That's R-E-T-O-O-L.com/sedaily. You can even host Retool on-premise if you want to keep it ultra-secure.

I've heard a lot of good things about Retool from engineers who I respect. Check it out at retool.com/sedaily.

[INTERVIEW CONTINUED]

[0:18:20.5] JM: Tell me more about the engineering problems that you were trying to solve when you started building visualization tools at Uber. What were the specific problems that you needed to solve?

[0:18:33.1] IG: When I started out, I was a part of building up a new team called marketplace health. We really dealt – I think was a magical challenge where we were looking at the core of the Uber business, which was really supply and demand and obviously, surge, which you can view where the demand outstripped to supply and prices were adjusted to balance the system, and which the view we took at that at the time was that it was more of an inflammation, it was a health issue, right? We thought about creating these tools that would able to help you visualize this issue in the system and then obviously, provide ways to correct it and get back to balance between supply and demand.

This is obviously involves big data. It's happens both in space and in time, so you'll be able to have a certain – you have a certain situation with supply and demand at a given time and you will be able to visualize that and you can visualize it on a map. You can make it as a heat map and you can see how the supply and demand, the supply is low in some regions and high in some areas. Also, the demand rights, which obviously is very driven by at the time of day, whereas people are commuting and the demand will be get into the city will be high in the morning and inversely in the evening.

Then also being able to see this over time, and so actually see how this evolves over time and then be able to do playbacks and then also be able to correlate things over time and actually, see how various types of actions that are being taken to improve this have long-term effects and be able to factor out other types of data that might be correlated that's masking the signal.

[0:20:13.3] JM: Right. You have this suite of tools that you ended up building called Vis.gl. Can you tell me about Vis.gl and the suite of tools that you built?

[0:20:24.5] IG: Yes. We started creating as we took the framework-driven approach. As we started out in one of these applications, we'd be a WebGL-based visualization layer system for rendering very large scale data. Then as we moved on being part of a visualization team, we were then all chapter was to help additional teams. We were able to take that same technology

we had built, that we had spent some extra time generalizing and then we had a few more for every team, every new application that we approached, we got a few more requirements for five new features that were critical. We took those and we spent some extra work implementing those in a way that they were also reusable and that we could move them into the framework.

This eventually grew out to an entire suite of frameworks. The leading framework work in this suite is deck.gl obviously, which is the data visualization framework, but it sits on top of other frameworks. We have a framework to deal with WebGL2 that we call luma.gl, which you can think of as the WebGL engine. We have very big framework for loading data. We have geospatial and 3D loaders for a wide variety of formats called loaders.gl and many others.

[0:21:35.5] JM: Deck.gl is this library for high-performance visualizations of large data sets. Why is it hard to render a large data set?

[0:21:47.1] IG: Well, if you don't need to render it quickly, if you don't need to do any changes, it's okay if the approach is a little bit slower and you can use any traditional technique. If you want to be able to seamlessly zoom and pan and maybe tilt and look at your data set from the other angles, if you want to animate it, then obviously and you want to have any type of fluid experience with it, then definitely you're going to need to use non-CPU-based techniques.

Obviously, things when you work on the GPU, things are more – they're more constrained. You get mess parallel computational power, but you need to get data in a very organized format. I think that's the challenge, the main challenge in this case.

[0:22:36.7] JM: Do you load the static datasets in bulk, or do you render streaming datasets? What's the typical input situation for deck.gl?

[0:22:47.0] IG: The core used case is that you load a single table and you load it in bulk. You load it atomically and you then basically display it. That works pretty well, certainly up to a million rows in the table, a little bit depending on what visualization you're doing and what machine you have. You might be able to push it to 5, 10 million rows. If you go beyond that and usually, loading might be acceptable fast for that use case.

Now if you go beyond that, far beyond that, you're going to need different approaches. We've actually done a lot of work on taking, going further. One very interesting trend that is happening now in technology is the development of tile layer approach is especially 3D tile layers. There's now a couple of standards, a couple of different systems for this. This is something we've done a lot of work on to build out, where you can basically represent a very, very big geometry, or very big set of data geospatially, using a hierarchy of tiles, or tables, if you will.

On the very high level, you would load an overview of the data. Then as you Zoom in, there are additional tiles with more and more data. It works maybe most immediately. The use cases are in a really big point clouds, or maybe really big photogrametry of big cities and so on, but the same technique can be applied to raw data. Using this, there's really no limit. You can go, certainly there's examples of city size point clouds where we have maybe a billion points, there's country-sized point clouds where you can go up to maybe 10 or a 100 times that and there's even continent-scale point clouds where you can be in the trillions of points. This is a very powerful technology.

[0:24:33.3] JM: When you think about a point cloud, like I've got a table in my database and this is going to be a point cloud that's going to be rendered on some geospatial map. There's so many engineering questions there, like how are you getting the geospatial map in the first place and then how are you superimposing the data on top of that map? Could you just take me through the loading of a visualization?

[0:25:02.2] IG: Yes, certainly. Well, when it comes to point clouds in particular, you could potentially just have the points in a table, right? You can have a very simple table. It could be say, become a separated value, so CSV or a JSON file. It could have just longitude and latitude and height of each point. Maybe you have some color on the pointer or something like that. It could be very simple like that, or it could be in the case of point clouds, there's sufficient mechanisms to do a significant compression, lossy, but significant compression. You also tend to want to store them in binary formats and so on.

You could use something like Drako compression. Then you're going to need a more sophisticated loading system to uncompress that on the client as it comes in. Essentially, it's just about loading in this – getting a really long array of with the coordinates of the points and maybe

the attributes of the points. Then you need to basically upload that one to the GPU and turn it into to attributes by a columnar table on the GPU. Then you have a shader which renders a point, but basically it has a little geometry, a little primitive geometry, a little circular square or something for each point and then a rasterizer and a fragment shader that basically turns that into a circle or whatever shape you want to have to represent each point. Then obviously, when I apply lighting or other types of things to make the visualization stand up.

[0:26:23.8] JM: Are there engineering issues that emerge from having too high of a volume of data? If you have a really big, big data set and you want to load that onto a map, are there any engineering concerns there? Do you just not hit any bottlenecks, you never have too many data points?

[0:26:44.2] IG: In practice, it has been remarkably rare that people run out of juice with the frameworks that we have. I mean, we do have a limit, practical limit as mentioned, somewhere between 1 and 10 million rows in a table. People just really tend to load a 100 million rows or really, really big tables. We've certainly done experiments. We're big believers in a technology called Apache Arrow. We do a lot of work on integrating and supporting Arrow. It's a binary columnar format, which really allows you to generate a table in one system in a server for instance and then completely without serialization you send it over a more or less memory map put out to the network. Then as we get it in on the client, if it's formatted correctly we can just basically upload it directly to the GPU without even touching it in JavaScript. There's a way to do batched and send our tables in a batched formats. You load them batch by batch.

By doing that, we've been able to load very big tables. We have exceeded 50 million rows. There are challenges obviously. You can get into a number of different issues. If you render that many elements and each of them, if you have two bigger radius on the points, you can generate the tremendous amount of points. You get an overdraw, where each pixel is rendered again and again and again. Once you're starting render and trying to render billions of pixels every frame, you'll get huge issues.

To really make a system that supports a 100 million use case and beyond as a single layer as opposed to streaming in tiles, there would need to be work, some additional work to be done. We have many ideas, but so far we haven't really had to go that far.

[0:28:21.9] JM: You mentioned Arrow. Last time I did a show on Arrow the way it was presented to me was that it was a data interchange format between Java and Python. Has it advanced beyond that, where now it's more of a general interchange format?

[0:28:37.0] IG: Oh, absolutely. I mean, if you go to the Apache or our github repository, you're going to find probably 15 different folders in the routes with different languages. You're going to have C++, Rust, there's JavaScript and there's Java and probably R and Python and a very long list of bindings. There's some very interesting technologies. There's also a in-memory server that allows multiple languages to share a memory that is organized, so you can basically have a Java program, a Python program, a C++ program all having a shared memory where they have an Arrow table and they can all access it in synchronicity.

That's unfortunately not possible in JavaScript on a browser, because we can't share memory with the rest of the system, because of the security concerns. But will obviously lead – it could be done in a node.js type environment and would obviously lead to fantastic performance characteristics.

[0:29:31.9] JM: Are there any WebAssembly things that could lead to being able to share memory across processes like that in the browser?

[0:29:40.3] IG: I don't see that. I think that that would be a basic a fundamental feature of the browser, whether it's supported shared memory. I guess, because of the massive security implications of that, it would have to be a special build of the browser or something. I'm not too hopeful that that would happen in the browser.

[0:29:57.6] JM: Got it. Deck.gl, this visualization library for large data sets. What problems did Deck.gl solve for you?

[0:30:07.8] IG: Well. I mean, on one hand, I think it provided a tremendous amount of efficiency, because of this solving, creating a very large suite of advanced visualization tools and being able to having to solve the same problems over and over again, it turned out, I mean, in hindsight, it's been a very good decision to take this platform approach. I mean, there's several

ways to look at the question, because one, it's obviously, this was used heavily internally, but it was also used externally.

I mean, there's maybe a separate discussion to be had about that, but the fact that we were able to get also external validation, external collaborations out of it was also adding value. I think it really allowed us to the fact that we are – were building this in a reusable way meant that we were able to lift the level of functionality quite a bit. We were able to take this framework much, much further than we would have done if we had created a number of separate visualization suits solutions for different applications.

[0:31:11.2] JM: The suite of tools of Vis.gl beyond deck.gl, it encompassed luma.gl, it also encompasses some other tools for data visualization, tell me about how these different frameworks fit together and what they did in concert.

[0:31:27.7] IG: Yes. I think it's partly up to maybe the architectural, or aesthetic preferences of the people who, including myself and other people built this and the design that we like is to have – you take one complex of functionality and you try to create independent modules, so that we avoid having too much coupling between different systems. Then we very focused then on each of these pieces being usable on its own. The way we chose to architect things as a number of different frameworks, each of which can be used on its own, but potentially if you used them together, they would work. You have the guarantee that they work seamlessly out of the box.

This was partly us, because it's good design, but partly also because we really wanted to encourage collaborations and we didn't necessarily want to make it an all-or-nothing proposition for users, whether they were internal in the company, or maybe more commonly external users. If you have something where you can allow people to choose the pieces that they need and replace the pieces that they whatever reason don't like, that makes many discussions much simpler.

[SPONSOR MESSAGE]

[0:32:47.0] JM: This episode of Software Engineering Daily is brought to you by Datadog, a full-stack monitoring platform that integrates with over 350 technologies, like Gremlin, PagerDuty, AWS lambda, Spinnaker and more. With rich visualizations and algorithmic alerts, Datadog can help you monitor the effects of chaos experiments. It can also identify weaknesses and improve the reliability of your systems.

Visit softwareengineeringdaily.com/datadog to start a free 14-day trial and receive one of Datadog's famously cozy t-shirts. That's softwareengineeringdaily.com/datadog. Thank you to Datadog for being a long-running sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:33:41.0] JM: As these frameworks evolved within the company, they became really sophisticated. They got to be a pretty considerable set of software. Did other companies adopt the open source frameworks once you open source them?

[0:33:55.4] IG: We certainly saw very big adoption of the tools in terms of the frameworks and in terms of simply github recognition and also in MPM downloads and those types of statistics. We had good interaction with a number of users, so lots of issues, lots of good discussions in github and our Slack channel. It was clear that there was a lot of – we were serving some need that there was a lot of interest in these things. Gradually with time, we've also done a few bigger collaborations, where major geospatial companies have chosen to collaborate with us for on various features and for various reasons.

[0:34:34.8] JM: Have you collaborated with Esri at all with the ArcGIS tool?

[0:34:39.4] IG: We have two major collaborations with S3. We've had the recent version of deck.gl. We launched the deck.gl ArcGIS integration, which allows you to use ArcGIS as a base map for deck.gl visualizations. Deck.gl does not contain a base map in itself. It provides map synchronized visualizations, and so traditionally we've had Mapbox and Google Maps between that. We now also have in ArcGIS integration.

The other big collaborations we've had with S3 is in 3D tile technology called I3S, index the same layers. This is one of the technologies that allow you to do country size to continent scale point clouds and you stream in tiles as you move the view frustum and as you zoom in the tiles that are needed to show either points, or for instance, photogrammetry of cities and so on. We've had a big collaboration with S3 there and we now have support in loaders.gl and deck.gl for indexing layers.

[0:35:39.7] JM: You're the co-founder of unfolded.ai, which is a spatial analytics and visualization company. This company was founded with the technologies that came out of Uber's visualization tools. What's the goal of unfolded?

[0:35:56.2] IG: Well, so I think it's a personal goal, certainly there's a huge passion for these technologies. Uber has been an incredible home for them and now allowed us to take these technologies very far. At the same time, we think even though we worked almost five years on these things, we think that there is so much more that can be done on them and I think we're just a way to see if we can take these technologies to the next level.

More from the business point of view, we do think that there is a huge need for new spatial analytics. There's more and more data out there that has geospatial information associated with it. We think that many of the techniques we have are really helpful. There's a lot of really good geospatial data out there, but it can be very hard to find and use unifying data, dealing with issues of data and integrity, doing join, spatial joins, temporal joins on geospatial data. There are many pitfalls. It takes a lot of work. We believe that we have the ability to provide a number of solutions around that that can make people's life easier [inaudible 0:37:06.5].

[0:37:07.5] JM: When I think about geospatial analytics, I think about the opportunity for basically, business intelligence tools, very useful business intelligence tools that could superimpose information on top of a map and give me insights, maybe even help me make decisions, like if I'm visualizing something like a shipyard. Tell me about the applications of geospatial analytic software.

[0:37:35.6] IG: Well, yes. I mean, in the general case, there a tremendous amount of use cases and applications. I mean, you could imagine for instance with a shipyard, you already you may

want to have a digital twin of the shipyard and you want to be able to on a computer, go in and see, understand how everything is located and placed and so on.

What we're doing from the unfolded side is more we're trying to look at – we're really looking at big data with two special components to it. Obviously, you want to be able to visualize that, you want to be able to have a rich toolkit of visualizations to match the data and what you're trying to – the insights that you're trying to get to to now allow you to be precise there. I think also, you need to be able to often, there's a signal in the data that you're looking for. Often, there's a correlated information. I think the typical use case there is it could be something like weather, for instance. Weather can dramatically affect the speed and which things are done in the real world, or the behavior of people, customers or whatever.

To be able to find a way to get access to that type of data and to be able to correlate, or maybe more precisely, de-correlate that data and get that, remove that noise so that influence from your signal, so that you can see the true underlying signal and compare over time and see how things progress. There's a tremendous amount of high-quality, more proprietary data being, or commercial data being available, made available now. If you look at some of the open data, there's just it's obviously census data. Census data has demographics and other things. Simply being able to take a data set you have and being able to correlate it with demographics can be very powerful. Population densities, or you can see whether especially at any area is growing, or obviously, other aspects about the average person that lives in a certain area.

[0:39:34.4] JM: We've talked to a few companies that are related to geospatial data. We've talked to SafeGraph and Esri, we've talked to Mapbox. Tell me about the landscape of geospatial companies and how they interact with each other.

[0:39:50.3] IG: That's a big question. Well, I think that there is obviously a number of data providers that are very focused. If you take something like SafeGraph, they're very focused on a certain types of location data. They produce really, really, really good data. Then obviously, they sell that data. How you then use that is a little bit up to the customer. Then there's a very wide companies, obviously S3 that has probably, I would guess, maybe hundreds of products and have something for everything.

Then you have most companies like Mapbox, they are very focused on providing base maps. They do that in enormous scale and worldwide scale. Then there are smaller companies that do consulting and helping companies make sense of their geospatial data, have some services around that. The total landscape is quite big. There's also satellite data. It's becoming I think more and more interesting. More and more satellite data is being generated, not only older static satellite data sets, but there are now satellites that are covering the earth with a certain frequency and you can get higher and higher resolutions that are basically updated more and more frequently. Being able to take advantage of that type of data is also an interesting I think opportunity.

[0:41:14.1] JM: When you look at the space of potential products that you could build within unfolded, what are you considering? Have you decided on what product that you're building it?

[0:41:23.9] IG: Oh, yes. We have. We're still in stealth, but we are getting close to having something out there. At this point, we're building a very generic tool to help people make sense of geospatial data. We'll stop there and then we will see if we basically, more specialized use cases as we –

[0:41:41.9] JM: How are you exploring the space of available tools to build? I know you have a services business, but can you just tell me more about how you're looking for opportunities, because there are already all these companies that exist in the space of geospatial data. Where can you find the actual opportunities that are untapped?

[0:42:02.9] IG: We think that there is a huge untapped opportunity in helping creating very easily accessible tools. Obviously, if you're a data scientist and you know how to program Python and you're willing to roll up your sleeves, you can do quite a bit on your own. We think that there's a lot of analysts that are either data scientist that would like easier tools, or analysts that could derive a lot of value and I think that that's a wide horizontal group, cohort of potential customers. I think that's a good starting point.

I think that too in every single industry, I think there will be a specific use cases. I think there is room for a lot of players and a lot of specific solutions in this marketplace. I would say that we have a pretty strong vision of what we want to build. I think a startup is usually good at doing

something. One, or very few things really well. I think it's probably best to start in doing the one thing that we believe with that we can do well.

[0:43:02.0] JM: Agreed. Just to close off, there are lots of ancillary technology trends going on right now. I think about satellite data, increase in satellite data, I think about improvements in the browser through WebAssembly, just 5G, for example. How are these different trends, or are there any particular trends that you anticipate changing the scope of applications that geospatial analytics could be applied to?

[0:43:35.8] IG: Yes. I think, I mean, just in general, I mean, I think the revolution that's happening I think what really deck.gl is capitalizing on is the power of the front-end. The traditional geospatial solutions are very focused on back-end processing for various reasons. We've focused on doing as much analytics and visualization as possible on the front-end.

If you look today, I mean, if you have a Macbook, you have a really powerful machine. It probably has multiple cores, maybe six cores, maybe they're hyper-threaded. You have probably one, maybe even two GPUs. You may have 32 gigabytes of RAM. That maybe considerably more than the virtual machine that's serving you in the data center. Really tapping into that power, I think is one of the – so maybe not the most dramatic industry trend, but I think it is a big shift in how we think about. Through this approach, we feel that we can deliver a very strong and superior user experience.

I do think that the – I mean, again, I think that the market for geospatial data is quite small. I think that there is some tremendously valuable geospatial data out there. It's very hard to get to, but there are all these companies that are working on this and making that data easier to access. I think that's certainly something that's going to happen. I think that and something that's going to maybe I would perceive as a shift when that happens. That means again, satellite data I think is as you point out, has huge opportunities.

I also think, I'm hoping obviously that the open source approach that we're taking to all of this also can help be transformational, because there's a lot of work today as the visualization technologies grow quite complex, there's new formats to do a 3D model visualization. You have

to implement something called GLTF, which of complex standard. You have to do implement support for these 3D tiles systems and other things.

It seems to me that some point this becomes a tax on the entire industry. If every company re-implements this on their own, then I mean, ultimately, the customers lose because these – as engineers, we could have been focused on doing innovative features and doing awesome solutions for specific use cases and we're all sitting and doing the same tables, takes things over and over again. I think the fact that we're now seeing this trend towards open source libraries, I also hope can really help improve the overall landscape tooling.

[0:46:01.7] JM: Okay. That's a great way to end it. Ib, thanks for coming on the show. It's been great talking.

[0:46:05.2] IG: Yeah, likewise. Thank you.

[END OF INTERVIEW]

[0:46:16.7] JM: When I'm building a new product, G2i is the company that I call on to help me find a developer who can build the first version of my product. G2i is a hiring platform run by engineers that matches you with React, React Native, GraphQL and mobile engineers who you can trust.
Whether you are a new company building your first product like me, or an established company that wants additional engineering help, G2i has the talent that you need to accomplish your goals. Go to softwareengineeringdaily.com/g2i to learn more about what G2i has to offer.

We've also done several shows with the people who run G2i, Gabe Greenberg and the rest of his team. These are engineers who know about the React ecosystem, about the mobile ecosystem, about GraphQL, React Native. They know their stuff and they run a great organization.

In my personal experience, G2i has linked me up with experienced engineers that can fit my budget and the G2i staff are friendly and easy to work with. They know how product development works. They can help you find the perfect engineer for your stack and you can go

to softwareengineeringdaily.com/g2i to learn more about G2i. Thank you to G2i for being a great supporter of Software Engineering Daily, both as listeners and also as people who have contributed code that have helped me out in my projects.

If you want to get some additional help for your engineering projects, go to softwareengineeringdaily.com/g2i.

[END]