



LECTURE TRANSCRIPTS

Internet of Things: Roadmap to a Connected World

Resource for registered participant use only

Contents

Week 1 – Module One: Introduction

IoT and the Connected World (Sanjay Sarma).....	3
---	---

Week 2 – Module Two: Architectures

The Architecture of IoT (Sanjay Sarma).....	14
The Web of Things (Tim Berners-Lee).....	28
Lessons from the Internet (David Clark).....	41
RFID Deep Dive (Sanjay Sarma).....	67

Week 3 – Module Three: Technologies

Network Connectivity for IoT (Hari Balakrishnan).....	81
Data Processing and Storage (Sam Madden).....	108
Localization (Daniela Rus).....	137
Security in IoT (Srini Devadas)	154
HCI in an IoT World (Jim Glass).....	173
Robots and Autonomous Vehicles (John Leonard).....	197

Week 4 – Module Four: Applications

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso).....	227
Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health (Dina Katabi).....	263
Smart Cities (Carlo Ratti)	282

Week 5 – Module Five: Conclusion

Roadmap of IoT (Sanjay Sarma).....	301
------------------------------------	-----

Week - 1 Module One: Introduction

IoT and the Connected World (Sanjay Sarma): Video 1 Part 1

Hello, and welcome to Internet of Things X. My name is Sanjay Sarma. I'm a professor of mechanical engineering at MIT, and I am the director of this course. We're very happy to kick off this course. Very exciting for us. Lots to talk about. So, let's get right into it.

So, why do we think of the Internet of Things, and what is it? And what I'm going to do is just kick things off by giving you a framework. And then, over time, we'll get into more depth. This first segment will also give you a sense of what the course will look like. And it's sort of a high-level business and technical view of the Internet of Things.

So, what is the Internet of Things? The fact of the matter is that, regardless of which industry you're in today, you've probably heard of Nest, which is a startup that was created by a bunch of Apple alums. And it became a big deal recently when, in 2014, Nest was acquired for \$3.2 billion by Google. And everyone stood up and said, whoa, a thermostat which is connected to the Internet? People actually buy it? And Google acquires it for \$3.2 billion? Why?

And I'm going to try and explain to you why this is the case. But one philosophical way to think about this is that, today, human beings generate a lot of data. But devices will generate data in the very near future, as we live in a world where the world sort of adapts to us and makes our lives easier. That is information as well, and Google wants to be in that space, as do a lot of companies. And this is the story of the Internet of Things.

If that wasn't big enough, here's a piece of news that a lot of people missed. You know, recently there have been a lot of recalls in the auto market-- Tesla, GM, et cetera have all had issues where they need to fix a car. Well, guess what Tesla did? Tesla was able to fix one of the issues with a straight over-the-air software fix. Why? Because Tesla cars are connected to the Internet.

So, the opportunity with this technology is enormous. And I will try and explain, try and convince you that, although there are challenges and there are issues, that to ignore this opportunity is not an option anymore. Now, any option, any situation like this, there are consulting firms or companies that will do analyses to figure out how big this opportunity is. And, frankly, the numbers surrounding the Internet of Things are staggering. And, even if you take a couple of zeroes off the end, you have to sit up and take notice.

So, for example, in 2015, just this year, there were probably about 10 billion connected things. There were about \$2 billion of services generated by the Internet of Things. But, if you look even five years from now, ABI predicts one quarter of a million connected cars. IDC expects about \$7 billion for IoT services alone. And Gartner expects something like \$300 billion from IoT products.

And you might say, well, that's a lot! But, think about it. If Samsung decides to connect every dishwasher to the Internet so that, instead of the dishwasher breaking down and then you getting upset, the dishwasher could tell Central that there's a problem. And the repair personnel, as they drive by your house, can say, hey, listen, I think I can preempt a problem. It would save Samsung money and save you money.

So, all of these devices will get recruited into this industry. And that's sort of what Gartner's talking about. IDC says that, by 2020, the global IoT market will be \$7.1 million. And ABI talks about 40 billion connected devices by 2020. Now, we're still talking about billions. But the number trillion starts making an appearance when we look 15, 20 years out. GE expects that, by 2035, IoT will have added \$10 to \$15 trillion dollars to the GDP. Cisco, not to be left behind, ups that number to \$19 trillion dollars. And ABI claims that, by 2035, there will be 450 million IoT-enabled cars. And these are staggering numbers.

IoT and the Connected World (Sanjay Sarma): Video 1 Part 2

So what is IoT? And I'm going to ask you, without building the animation yet, to do a little thought experiment. Go to a 12-year-old child and say, young man, you see that switch? And the child will look at that switch. You see that light bulb? The child will look at the light bulb. And say, how is it when I hit the switch, the light bulb comes on?

I want to pause here for a second and I want to you to do a little thought experiment. What do you think the child will say? I've done this experiment dozens of times. And you know what the child almost always says? They say, Wi-Fi, Wi-Fi. The switch tells the bulb, and the bulb turns on.

And you laugh because it's cute. And try and draw out the actual system we use today, which I did. And so this is my drawing. And explain to them, no, no, no. That's not what we do. What we do is you get power with a hot line and a common line. Comes into the house. Goes in through the walls.

Come to the switch. The switch actually breaks a connection, or makes a connection, and then we snake that same wire all the way to the light bulb. Now think about it. Who was smarter? Was it the 12-year-old or you?

The fact of the matter is that the way we connect things up today is extraordinarily primitive. And to move that switch 10 feet to the right in an office building might cost several hundred dollars because of construction cost, electrician cost, et cetera, et cetera. But if you had Wi-Fi, which is, I don't know, a couple of bucks on the switch, a couple of bucks on the light, then the switch could be on a Velcro. And you go unstuck it and stick it somewhere else. And that's it.

So for a few dollars, in principle, IoT will then untether the switch, as opposed to baking it into the construction as we've done today. And that is the world we live in and we take for granted. So what happens with the Internet of Things is it opens up an entire vista of opportunities that you have to sort of sit back and say, my god, that is amazing. Right?

And some of it has crept up on us, and we sort of take it for granted. But a lot of the stuff that we do today, for example, if you've used a Zipcar, a Zipcar-- well, it was actually the first example, at some level, of a connected car. But that was enabled because you could take a car and you can book it over the Internet and unlock it.

So this is an incredible opportunity for all of us, and something we should take very seriously. And frankly, the question you've got to ask yourself is, is there anything in the world that won't be touched by the Internet of Things? And it's actually hard to think of examples.

I mean, only five years ago, if I said to you that the taxi industry would be disrupted by the Internet of Things, we would have laughed. But yet, here we are in the age of Uber. And cities are going to change. And we have, in fact, a talk coming up later about cities.

Cities where something like several percent of the traffic in any city is cars looking for parking. And just imagine if a parking meter could alert you when it was free. And you could just go book it. And with five minutes of a shop, you have the parking meter.

The opportunities for electricity, water, waste management in cities. I won't tell you about each example. I'm just going to go through the extent of the examples so you have a sense of how enormous this opportunity is.

Buildings are going to change. HVAC, everything. Homes are going to change. If you have a Nest or a Dropcam, your home is already changing. Kitchens are going to change. Your dishwasher, appliances, microwave ovens. The toaster on the Internet may be something that may or may not happen in the short term, but a lot of other appliances will change.

Security is going to change. In fact, I look at security companies today and I wonder why they haven't taken this opportunity by both horns. But it's a huge opportunity. And home health is going to change, especially with an aging population. And that's just still starting at cities and working my way down. Right?

Take factories. Factories are going to change. Look at the machines in the factories-- they can be instrumented. Robots-- you've heard of the robotics evolution. More and more they'll be connected to each other. There's, in fact, a whole effort now called cloud robotics which speaks to that.

The people, from a safety perspective, they may be instrumented-- just make sure it runs safe, right? The supply chain will change. I played a key part in the development of modern radio frequency ID. And the implications of visibility of the supply chain on the operation of factories and retail is enormous.

So we can go on and on. Transportation, airports, ports, stations are going to change. Roads-- why should we pay a toll with an RFID tag? Imagine if I could just pay with my cellphone. So toll's going change. Railway stations.

And then cars. We've talked about trains, planes. If you could take the tragic case of Malaysian Airlines MS370, which was lost, and only now are fragments showing up, that could have been found so much more quickly if, in fact, it were truly connected to the internet.

Traffic could change. Traffic management. We could track emissions. And you could go on and on. Will my chair ever be connected to the Internet. Maybe not, but there may be reasons to do that. Hospitals. Equipment tracking, staff tracking. Where's the doctor? There's an emergency. How do I reach the doctor?

Patient tracking already happens. So for example, in the neonatal wards now, little babies have RFID tags. Ambulances, trauma. One of the big things that happens in medicine, in surgery, is keeping track of equipment. For example, scalpel, sponges, et cetera. A huge

industry, actually, in pharma, medicines. That means consignment selling of medicines, keeping track of when someone picked up a medicine or a stint, coaching tracking of medicines, and so on.

Even something like sports is going to change a lot. Already, and sort of peripherally, and I've been watching this, that there are now efforts to track players in football so they can tie it to fantasy football. If you, for example, watched equipment such as Formula One cars, the amount of telemetry that comes out of a Formula One car is just staggering.

But pretty soon, everyday weekend tennis players might have a little thing that measures how their stroke was. Golf players may have something similar. Sort of the Fitbit thing moving to equipment. Ball tracking, player tracking, performance telematics. All this is going to change.

Now, that's just the start. Now you look at big industries, whole industries, heavy industries. Oil and gas-- could we have seen something coming before the Macondo disaster? Mining-- just an enormous opportunity. As you may know, mining involves many, many piece of equipment, their safety issues, their chemical leaks, et cetera. And the opportunities with connecting things up are amazing.

Shipping, container tracking, et cetera. Huge opportunity. Construction-- this is something that's going to happen in the near future. Construction is going through a revolution right now. And you will see not just IT, but IoT pervade construction. Power, grid monitoring. If you'll recall, the big power loss that New York suffered a decade ago. Just imagine what sensors in the power grid could do.

And water supply monitoring is going to change. I just want to give you a sense of how extensive this is. And in fact, I encourage you to look around you and ask not where IoT will be, but where it won't. I mean, pretty much anything you see, you might want to ask yourself, will that go on IoT?

IoT and the Connected World (Sanjay Sarma): Video 1 Part 3

One of the things I also want to clarify is there's a subtle difference between the Internet of Things and connected things. Connected things is when things are connected to each other, but they may not be connected to the Internet. Internet of things has been things that also are connected to the Internet.

But that boundary is being blurred very rapidly because most things are sort of connected to the Internet and a lot of things are connected to each other. And so pretty much all these little islands that we already see, like your car has 200 senses. The moment something on it is connected to the Internet, you're in the realm of Internet of Things.

So I put up a couple of pictures here, an airport, a train. And you can imagine how IoT enables new ways of thinking about systems. So for example, if a bunch of flights arrive at the same time or in a clustered way because of weather delay, can the taxi drivers be told of that? Can the subway be informed of that? Can crowd control be put in place?

So the other thing about Internet of Things is how things will get connected, connected with each other. Every day we hear about things like self-driving cars, SDVs. You'll hear more about it later on from another presenter. Self-driving cars, trust me, will need to be deeply connected to the internet so they can collaborate with each other on things like traffic, on hazards, etc. So all this is happening very rapidly.

So my last slide in this section is just to say that it's such an overwhelming concept. And that's also part of the problem with the Internet of Things. And during the course of this presentation, not just mine, but my colleagues as well, you'll hear about some of the challenges with Internet of Things.

But one way to think about it is that there are core component technologies. And examples of that are location, indoor location. We know how to do outdoor location, but often you need indoor location. How do you do that? Sensors.

A lot of the Internet of Things is about sensors, which is how do you sense the world? Interfaces, how do you interact with the world? I mean how do you interact with all these little devices? Set-up is going to be a big problem. How do you set these things up?

And then actuators. Actuators are things that actually impact the world. So for example, if the temperature sensors says the temperature's dropped below a certain threshold and the owner of the house is coming home, turn the heating on. The turning on is acting, and the way you do it is through an actuator, which is a little valve that turns on the heating, a switch.

So there are many aspects of actuators. We can't go to them all. That motor's an actuator. But in particular, robotics and vehicles are two very important actuators. So you'll see some focus on that in this presentation.

Then there are systems issues, and I'll speak to that. Use networking. How do you do networking? There's hundreds of standards for networking these devices.

This is architecture. Professor David Clark talked about architecture, a very profound, important topic. Because he talks about the history of the architecture of the internet, and we need to learn from that. And I'll talk more about it.

Data management. We're going to generate lots of data. How do we handle it?

Security. Huge challenge. Security and privacy are two very, very important topics in the Internet of Things.

And then there are Applications. So that's the top piece of this. Smart cities, smart buildings, I've talked about all of them. You'll see more on smart cities and smart buildings, but you really can't cover the scope of the applications in this course, but you'll get a taste of it. So thank you very much and you'll hear more from me shortly.

IoT and the Connected World (Sanjay Sarma): Video 2

So now, the question I want to talk about is so what. Yes, it all sounds nice, but how do I get going. I'm collecting the sensor data, but what do I do with it?

How do I act on it? How do I build intelligence? How does databasing fit into it? And do I keep the data? So we need a mental model for that, and we'll present that now.

What I say to people is, the most powerful model to use and the model that we are all most comfortable with is the human model. The human model is one of sensing and acting. Sensing and actuation is the technical term. We have neurons, and we have muscles. And in between, we have a nervous system that takes all this data and processes it.

And if you think about the human model and then apply it to the Internet of Things, what we're saying is, you have sensors, you have actuators, and you have a reflexive action in between. The great thing about a reflex, and the reason we call it a reflex, is reflexes are unthinking. They're instinctive.

So if you touch something hot, your finger doesn't have a teleconference or video conference with the brain. It just moves. That's because it's a reflex. We do things like breathing and so on with reflexes.

So sensors collect, in the Internet of Things, data about the world. Like with human beings, a very key part of data is identification. I want to know what that thing is. And that's where RFID comes in, for example, of barcodes. We get location information, indoors by the way. That's surprisingly difficult to get location information.

You get environment information. What's the temperature? What's the pressure? Do I smell a gas? And physical state, so things like, is the thing hot, is it on, is it off, things like that.

And then that information now goes through a reflexive sort of control system, if you will, and then goes to the actuators. And the actuators in the Internet of Things broadly have two types. One is display, which is I'm going to display to you-- as you are the human in the loop-- I'm going to display some information.

The other actuator is physical. I have a motor. I have a robot that moves. It's something automatic that happens. But that's reflexive.

But actually, what about thinking? Don't you have to learn from it? By the way, one thing I want to tell you is that all the stuff I'm sensing has been accelerated by the enormous, the incredible advances that cellphones and other technologies have created.

So for example, RFID, again a technology I'll talk about more-- allows sort of ESP, Extrasensory Perception, of ID. Barcodes are sort of a fiducial image recognition, is very

powerful now for recognition image processing. Wearables, making pervasive sensing possible.

Radar? Yeah, yeah, radar. A lot of cars have radar now. They can detect 3D stuff. And then all sorts of physical, chemical, and spectral sensing technologies that are not available.

And then wireless makes it possible to build all these sensors and connect them up. Wireless has become so cheap, you can embed wireless, like in a light switch and a light bulb. By the way, the Philips Hue can be used that way.

The same wireless now allows sort of a local reflexive controller to also talk to the actuators. So it can tell an actuator turn on. It can tell a particular heat vent to open and so on. Or I can tell a display-- even displays have become so cheap, because of things like E Ink, and OLED, and so on.

And then finally, because of batteries, because of MEMS technology, because of the miniaturization of motors, actuations become really inexpensive. So you can turn a valve on very inexpensively. But that's reflexive.

But if we look at the human body, there are actually two more layers in which actions occur. There's the reflexive stuff. Breathing, I don't think about it. It just happens reflexively. If I touch something hot, my finger is going to move. It's a reflex.

But then there's conscious stuff, which is when I touch something hot, my body moves my finger away. And then because it takes a little more computing, my brain says, hey, wait, wait, wait, why was that hot? And oh, the stove was on, or the oven was on. Well, who left it on? Well, let me turn it off. So that's a little bit more conscious.

But then the next step is cogitative thinking. So the cogitative part is, you might say, wait, why was the oven on? And then you have to sort of recreate a story. And go, oh, you know, my daughter, she just learned to bake cookies, and she baked cookies. She learned to use the oven, she did it well. But she forgot to turn it off.

And so now, you form sort of a more long-term thing, which is, the next time she uses the oven, I'm going to remind her. I'm going to have to check after she uses it. So that's sort of the big data, sort of analysis to look for patterns.

So the finger-- actually, your finger does not record to your brain every touch it encountered. So one of the things about this is figuring out what you want to retain and what you don't. And it's only when you touch something hot that that whole chain triggered. And that's called exception-based processing.

So the key thing about the Internet of Things is not to fall into the trap of, I want to record everything. No. It's you want to record the things that make it up for cognition

and, to some extent, for conscious thinking. But reflexively deal with data as it comes and not keep data that you don't need. Otherwise, you'll just flood the system.

We can now move onto a more clear picture. We can drop the human body analogy. And now what we have is sensors and actuators. And we can start thinking about how to implement it.

You know, when you do something cognitive or conscious, now the compute cycles are much quicker. And so for that, you really need to go to the cloud or go to a computer. But the reflexive stuff has to happen instantly.

So for example, if a fire sensor detects smoke, the alarm must go on instantly. So what you need is sort of like the human body. The spine is where the reflexes go through. And so you need that to be happening in real time locally.

But the stuff that happens in the cloud can be a little bit less real time but involves more compute cycles. And what the reflexive piece must do is send up to the cloud the things that are worthy of further consideration. And that is what is sometimes referred to as fog computing.

You can think of the reflexive piece, the controller that is local, as a piece of the cloud, like a proxy of the cloud, that is down right at the action signaling real time action. So for example, if you hit the brake on your car, you don't want the pedal to talk to the cloud before the calipers close. You want a local controller.

So think of the reflexive piece as a local proxy of the cloud. And what do we call the cloud when it touches the ground? We call it fog, right? So that's one way to think about fog computing.

And that's where the real time piece comes in. And that's where the non-real time piece comes in. Fog is-- the cloud part is not real time, depending on the application. For some applications, it's real time enough. But for a lot of true, hard, real time applications, you need a local controller.

Now, another issue that comes up with the Internet of Things is, well, what about external control? So what is external control? So your home has an Internet of Things system. But let's say that your car, you want to program your car to say, listen, when I'm near home, I want to play this music, I want the lights to come on, I want to unlock the door, and I want the heating to turn on.

Well, that means that instructions may come from the outside. The problem with that is that those instructions go to actuators. And that also creates a security problem, because what if some of those instructions are wrong-- unlock the door, but you're not home. So security, no matter what you do, is always going to be an important issue.

One of the ways to deal with that is to create, what I call-- this is not a term that is in wide use-- but I call a cognitive firewall. And what I mean by that is you want your system to have a model of itself. And you want the system-- and to use the human analogy-- to have the good sense not to do certain things.

So for example, if an external system says, hey, listen, turn the lights on. It is dark. Well, the downside isn't that great. And you can look at the time and say it probably is dark, sure turn the lights on. But if the external system says, I want you to turn the microwave on for 100 minutes, we need an internal model that says, hm, no, I don't want to do that, I'm sorry. That's what we humans do.

And so we need some element of cognitive capability. If you're going to take external instructions on actuators, you want very tight security around that. And you may want someone locally to say, yes, to act as a proxy for that system. There's a whole lot of mathematical modeling that can help with that. And we have some stuff that we're working on in our lab that uses something called observer theory and estimation and so on to do a lot of that stuff.

So in summary, just to end this section, I'll say that IoT is local, but it's also global. Your home, automation is great. But if you can control it while you're on vacation, that is great. IoT is sensors and it's actuators. It's both. It's not just sensors, it's not just actuators. You need both.

Some IoT is real time and some is not. And the stuff that is real time, you may want execute real time. But the stuff that's not, no reason to execute real time. And you may be able to use certain resources from the cloud.

This is a perfect place for machine learning, by the way. And not all data needs to be stored. But the most important thing to take away from this is security is going to be a key issue in the Internet of Things.

Week - 2 Module Two: Architectures

The Architecture of IoT (Sanjay Sarma): Video 1 Part 1

I'm now going to tell you the story of Radio Frequency ID, RFID, something I played a central role in, but also something that is considered today one of the earlier implementations of internet of things. But before I do that, let's just quickly look at sensors in general. What are sensors?

You can actually classify sensors in many ways. You can classify sensors by what you measure, also call the measurand; the acoustic signal strength, for example; or the chemical concentration; or the optical signal, which is sort of what cameras do; or the magnetic strength.

You can also look at it sort of more fundamentally. You know, you don't really measure certain things by themselves. You measure something else, a proxy for that. And usually it's something like mechanical displacement.

So, for example, a weight sensor doesn't actually measure your weight per se, it measures the displacement of a spring. Right, it converts it to something else. Or a voltage, or a chemical reaction.

You can classify sensors by the transduction, which is how you convert one thing to another. So for example, you might use the thermoelectric effect or you might use the photoelectric effect where the photons create an electric effect. Or a photoelastic effect where elasticity results in a transmission or polarization of light.

Or you might look at a high level function. So, for example, you can say this thing sensors ID, this thing sensors a gesture. By the way, sensing of human gestures is what interfaces are all about.

So typing is a form of sensing. Voice is a form of sensing, sort of a higher level sensing. Or the sensor senses an explosive device, which is, of course, of increasing importance in this day and age. Security is a big IoT application.

You can also look at sensor modalities. So, for example, there are static time-series sensors. So, for example, a temperature logger. There are 2D arrays. So, for example, imaging in electro-optical, which is sort of visible light, or infrared or ultraviolet, say.

There's 3D sensing now. Kinect or radar detect 3D. There are mobile sensors. Your phone is sort of a sensing modality now, sort of sitting with you all the time sort of sensing stuff.

There are scanning sensors. So, for example, Google Street View scans, and that's what this image shows, scenes from a car. So that's scanning.

And then there are fiducial sensing. Fiducial sensing is when you have, for example, a homing beacon, right? You're actually sensing not nature, but you're sensing something you put in there just so you could sense it. It's like a lighthouse.

So let me know, with that, plunge into the RFID story. RFID has been around for a long time, but RFID is an RF-based fiducial ID sensor based on RFID tags with static or mobile applications. And usually it's a time series or a membership set, because an RFID reader can read multiple things at the same time.

And this is actually a picture of an RFID tag that is at Macy's. And it has the [? term ?] on it, the Electronic Product Code, which I'll tell you in a minute. But the story is I was at a Macy's and this was my first encounter with an RFID tag, an Electronic Product Code, EPC tag, that came out of my lab. And I'm going to just show that here.

This is the EPC, Electronic Product Code. And I asked the Macy's lady if I could have the tag, and she said, only if you buy the shoes that come attached to it. I actually bought the shoes, but I framed the tag.

The Architecture of IoT (Sanjay Sarma): Video 1 Part 2

So I just want to go through the history of RFID. We did not invent RFID. RFID was invented in 1948. By 1974, it was being used in the Sydney Harbor bridge. There are many people who played an incredible part in developing this technology.

But in 1998, several of us, Kevin Ashton, David Brock. Kevin Ashton is from Procter and Gamble, David Brock is at MIT, research scientist. We started looking at RFID, and we created something called first the Distributed Intelligence System Center, and then the Auto-ID Center. And we received funding from what was then Uniform Code Council, which is now called GS1, and also from Procter and Gamble, from Wal-Mart, from Gillette, and that's at the center got going.

And over time, we've had more than 100 centers. By 2001, we developed the first standards. One of the problems with RFID is there's a proliferation of standards. And as I'll talk about later on, you want to converge standards. And we developed new standards working with dozens of vendors. And we presented our full standards.

In 2002, Gillette ordered half a billion RFID tags from Alien Technology, which was a startup company. That shook the marketplace. Ironically, that delivery I don't think ever took place, but it just sort of woke up the market. And in 2003, Wal-Mart mandated that over time, all its incoming inventory would have RFID tags in it. There were more mandates in 2004.

In 2005, the first bulk tagging started taking off and used protocol that came out of a follow up to the Auto-ID center called EBC Global that was not for profit standards body, created this new standard working with vendors called Gen 2. And by 2010, Wal-Mart was claiming to tag all men's innerwear, socks, in all stores.

In 2013, there were about 4 billion RFID tags. And by 2014, I saw them show up at Macy's, and that's the picture I showed you. So let me give you the hypothesis that drove our approach. Before we came along, RFID tags used to do a lot. They stored data. You could write a lot of data to them. And the result is that the protocols had become very specialized, and the chips had lower functionality than they probably didn't need. And as a result of that, the chips became big. So they were expensive. And moreover, big chips consume more power, so they drew quite a lot of power.

Passive RFID tags have no batteries on them. This came inspired from the readers. If the chips required more power, then this range dropped because they can't be much closer to the reader to scavenge that power. And when the range dropped, the number of applications that could use them dropped.

And so it created this vicious cycle of specialization and extra functionality in chips. So we came on and said, stop. Let's stop doing that. We said, let us put only a license plate on the tag. That's it. And let's just read the license plate. And we call that the electronic

product code. And so this is the old version of the electronic product code. It's changed a lot, but I just put up for historical reasons.

And that's where EPC comes from. But thank heavens we were foolhardy enough to jump into it, because that is the state of the industry today. But by skinning down the chip, we were able to solve a number of problems. Now that created some other challenges as well. So for example, if you make the chip so small, how do you pick it up? How do you drop it? And we said, look, over time it will be like printing. And we did some analysis of chip costs and printing costs. And we showed that that was possible.

So one of the issues that happens when you make chips that small and tags that small-- remember, a tag is a chip with an antenna attached to it. The question is how do you manufacture on a mass scale? And we did a lot of analysis on chip manufacturing, tag manufacturing, we worked with a number of very, very innovative companies around the world. And we were able to sort of convince industry correctly that RFID tags could be applied, and that's the word applied, to products using a printing-like process.

And so we did a lot of work on that. I actually ended up attending a course on printing in the paper industry to understand that. But wait, people said, so where do they put the data? And we said, put the data? And this was before the cloud, the word didn't exist. I used to point to the sky and say, put it in the sky. And people would say, how would you do that? And I would hold up my old Motorola brick phone and say, trust me, one day this will be on the internet.

And a lot of people didn't believe us. And we barely believed ourselves. And so we said, we will move all the data through the Internet. It's sort of like if you've got a speeding ticket, the officer doesn't actually write your speeding ticket on your license plate. They write it into a database, which they can access from their computer, right? So that's solely the approach we took. And that's how RFID finally really started taking off.

So the market has grown pretty tremendously, \$4 billion to \$5 billion option to RFID tags in 2014. The growth is expected to really increase in the coming years. Health care, \$5 billion by 2022. The total market is expected to be, by 2022 about \$15 billion. My prediction about the breakdown of the revenues is that 30% of the cost will be tags. Tags, I think will get somewhat commoditized. 30% will be readers, and the rest will be software services and solutions.

I think in fact, we've already seen entire IT systems will have to be rewritten. Because the visibility that RFID tags give is amazing. But one of the things we've discovered with RFID over the years is it's sort of like a doctor in the modern age, treating patients without x-ray or MRI. And when you can start seeing what happens to your inventory, the insights are tremendous. And the level of control, the high resolution management you can execute, is really quite amazing.

The Architecture of IoT (Sanjay Sarma): Video 1 Part 3

So where will RFID take off? You know, there are lots of industries. And here's sort of a layout of all the things that could happen. Item level tagging is sort of the ultimate, where everything is tagged. There's backroom, the supply chain, and then whole other industries, medical, transportation, et cetera.

There's a big concern about privacy with RFID. I think that's a valid concern. But it was a little exaggerated when RFID came out. But you have to take privacy very seriously in the Internet of Things and with RFID.

I ended up testifying to Congress to a subcommittee on privacy. And so we had to put in a lot of things in RFID in the electronic product codes, including choice, and notice, and so on. So if you buy something with an RFID EPC tag on it, it's says EPC, as I showed you the price label on the product that I photographed.

But you know, it's a strange thing. And I just want to make a point here for Internet of Things, as well. The thing that really took off was not these other things that we thought would drive it. It was what we thought was the most difficult thing. It was item-level tagging. And the reason it took off there is because that's where the business case was strongest, apparel tagging. And that's why my pair of shoes came with RFID attached.

So if you haven't read a book by Geoffrey Moore, called Crossing the Chasm, I strongly recommend it. And this RFID story actually picks up on a lot of the things he talks about. RFID did not take off until several things happened, though most important thing was we had to identify the ideal market. And that's apparel.

And the reason is, if you look at a rack full of coats, they all look the same. But some are different size. Some are subtly different colors. It's hard to tell them apart for the store manager. But when you have RFID, it's almost like ESP. You know what inventory is there. You know if the standard sizes are there. And the applications are enormous for an industry, which is frankly a little bit backward. Apparel is sort of an old-fashioned supply chain.

But the other thing that happened, very interesting, I thought that the 2008 crash would actually destroy the RFID industry. It did the opposite, because all these companies went bust or were in trouble. And they're holding all this inventory, which is radioactive. So the importance of reducing inventory and managing inventory became very clear. And RFID took off as a result of that. And over the years now, companies like Macy's, which is a big apparel producer, have expanded their use of RFID.

One of the other stories of RFID standards, as I said to you earlier, before the Auto-ID center, RFID industry was a mess of standards. But working with a number of companies, we created a suite of standards under the electronic product code banner. And if you do a search, you'll find a bunch of these white papers that I and my colleagues wrote.

In 2001, we released the first set of the standards. They've got Class 0 and Class 1 standards. I wrote a roadmap that Dan Engels on the future of standards in RFID, which has largely been agreed to by industry. In 2004, a new standard came out. It's called a Gen 2 UHF standard. I won't bore you with that. But there's a standards document you can read.

It has since become an ISO standard. It's an 18,000-6c standard. RFID is subtly different from communications because the tag is really dumb and has no power. So for the reader to send it information and get information back, and especially when there are more than 100 tags or almost 1,000 tags, and especially in the multiple readers vying for these tags, it's a technically difficult problem. And so we wrote a lot about it.

And the standards, or the new standard is a brilliant optimization of how to do that. It's, as I said, an ISO standard. And EPCglobal, which is a follow-up standards body we created. And as a disclosure, I was a chair of that board until recently. Also released, standards for readers, and for data sharing, so there's a full suite of standards.

And here are some publications if you're interested in the technical problems. I also recommend looking up, if you're interested in the detail, the ISO 18,000-6c standard, which in my world is referred to as the UHF Gen 2 standard. Thank you very much.

The Architecture of IoT (Sanjay Sarma): Video 2 Part 1

I'm now going to talk about the importance of architectures, and this is something I believe that we are not paying enough attention to in the Internet of Things space. I've put up this picture, because when you see an electric grid, I'm hoping that you will think architecture. Whole industries take off when there is a simple dominant architecture, and sometimes the architecture is counterintuitive.

And examples of industries are alternating current, AC, the power industry. Another architecture is packet switched networks. We all know a lot about that. Cellular telephony is another one. There's actually an architectural choice that draws cellular technology, which is embedded in the term cellular technology. The worldwide web-- Tim Berners Lee is going to talk later. Amazing stuff on architecture.

Consider power grids, for example. Alternating current was not intuitive, but it really makes possible the power grid that we take for granted today. So if you look at the power systems today, you have a plant-- coal or nuclear, whatever-- and they generate electricity. But what we are able to do is because of AC, we're able to step up the voltage to high tension. That's what's called a high tension wire.

And so with much lower current, ship it long distances, and that reduces resistance losses. It's because resistance goes with [INAUDIBLE], so the current is much lower. If you had to ship current over long distance without increasing the voltage, you would lose so much to transmission, and the impact on environment, et cetera would be spectacularly bad. But ultimately, current was not intuitive.

And by the way, the other thing I should say about alternating current is alternating current not only enables step up and step down, but it's also very compatible with rotating machinery. But AC came from a fight between Tesla and Edison. That's why I was very happy to see Elon Musk name his company Tesla. Edison wanted to push DC, because he wanted to sort of rule the world, and Tesla almost lost the battle. And Edison attacked him intellectually for his efforts. And anyway, AC took off, and that's a fundamental, fundamental architectural decision.

Cellular telephony is similar. If you are old enough to remember, we had walkie-talkies. The way walkie talkies work is my walkie talkie talks to your walkie talkie directly, but even if I'm in the same room as you, and I make a cell phone call, obviously, my cell phone doesn't talk to your cell phone. It goes through the tower. It actually sort of goes through the cloud, which please remember, because I'll come back to that.

And so what happens in cellular telephony is that you have cells with towers, a cell tower, and when you are in one cell, and you call another cell, even if you're both on the same cell, your phone talks to your tower. It goes to home base, and then it comes back to the tower, and then that talks to your phone. So this is actually simplification. I want to make the point that there was a pretty fundamental architectural decision to move away

from talking directly amongst peers to talking through the cloud as it were. I'm misusing the term cloud.

But the worldwide web is an extraordinary example of that. Great architects are not recognized in their time, although you will hear from Tim Berners Lee, who fortunately was recognized in his time. What did Tim Berners Lee do with the worldwide web? He said, it is not a continuous session. You open a page. You click on the link. It goes to another page. And in some ways, he skinned out a lot of the functionality of the Internet, and that's how the worldwide web took off.

It's a very simple design metaphor. My kid understands it. That is very fundamental. The second is standardized interface. You need standardized interfaces for reuse and reconfiguration. Not just the first time, but the second time you want to do something-- that's when standards help particularly. You need abstraction and modularity. You need to prevent cascading failure. You don't want things to be so bound together one thing breaks, the whole thing breaks. The worldwide web has elements of that. If the server is down, the client doesn't break. It's not a direct connection. It needs to enable repair.

Most good architectures have an element of what is called an hourglass, which you will see repeated not just in my slides, but you will hear it from David Clark later on. And they must be simpler to secure, and they must take care of things like privacy. And good architectures are often counterintuitive.

So an hourglass architecture is sort of how the Internet was tamed. There are lots of ways-- wires, fiber optics-- at the lowest level. There's all sorts of standards at the level above. Your cell phone has one standard. There's something called SONET, et cetera.

And then there is local ethernet, all that stuff-- PPP, Wi-Fi, all that stuff. But the thing that makes the Internet work is everyone agrees to Internet protocol, IP. And right above IP is TCP and UDP. The fact that those two are so standardized, it skinnies out the standards. It doesn't matter if the scale's below. It doesn't matter if the scale is above, but it skinnies it out. If you agree to that, everything else can be made to work. How it enables all these amazing applications, we take for granted today-- the worldwide web, email, telephony, video, et cetera, et cetera. So that is what an hourglass architecture is.

The Architecture of IoT (Sanjay Sarma): Video 2 Part 2

Unfortunately, a lot of the Internet-of-Things implementations today are truly, what I would call, a cobweb of things. They're really a bit of a mess, or they're implemented sequentially, or you might have someone come in and connect your light to your switch using Bluetooth LE. And someone else says, I'm going to connect your thermostat to something else, using ZigBee, and someone else uses six [INAUDIBLE] and so on with several different protocols. There are different bridges and patchwork routers. And often security's an afterthought.

That's a problem. This is a very serious problem. Maintainability-- my god, it's bad enough to build the first system, but what happens when the poor person who has to fix the system comes in or has to extend it? I mean, it's a patchwork. It's sort of like what we used to call write-only code. You can't read it ever again because it's so difficult. And one of the things is, remember, I talked about reflexes? If I touch something hot, my finger needs to move.

So there are rules that you need. And all the rules are baked into the hardware and into the way it's wired. So to change the rules becomes extremely difficult. In fact, I did an IoT implementation in my house a few years ago. And recently, I just unplugged the whole thing. Two reasons-- one, I was worried about security. And the other was the rules are baked in. It was very hard to sort of rethink them.

So the way I think about it is that the future is through the cloud. And the way it would work is that you lift everything by creating avatars in the cloud. So you have a switch; you create an avatar on the cloud. You have a light bulb; you create an avatar on the cloud. So there's a virtual switch, and there's a virtual light bulb. And you do that securely. So when you hit the switch, the cloud switch gets turned on, right?

Now, obviously, for real-time reasons, you may want to go to the cloud every time. What if there's a storm and the Internet broke down, right? So you may have a local representative of the cloud, fog. It's like your reflex action, which takes those rules and implements them on a run-time basis. So it at least gives you real-time action and it's secure from a disaster. So you may need a fog.

But the philosophical way to look at it is you're replicating things from the real world in the cloud. And then you pull all the rules out, and they're abstracted in the cloud. So you can say, well, if I come home, turn the lights on. Or you might say, if I come home, and it is dark, turn the lights on. So it's easy to sort of manage them. And that's really the way the world will go. And the communications across things can go through their respective cloud, sort of an "intercloud," if you will, right?

So, for example, my car would have its own cloud. My home would have its own cloud. I can program rules across clouds. Hey, car, tell the home that I'm nearby. Hey, home, when you see the car, turn the lights on.

We have to skinny down what is a somewhat chaotic IoT architecture right now. Instead of trying to skinny down and saying, ZigBee wins, because that doesn't necessarily solve this problem, what you do is you say, look, I don't care. You different devices. They speak different sort of semantics, different protocols. Some people use ZigBee. Some will use Bluetooth. Some will use other protocols. There'll be bridges, routers, controllers. Just make them all cloud things.

Just bring them to the cloud. And then we will interface across the cloud. And the reason this works is software interface is always much easier to fix than hardware interfaces. And the good news is we know how to take things securely to the cloud. If you haven't seen, Amazon sells these little buttons called Amazon Dash, with, for example, a Procter & Gamble Amazon Dash. It's a Wi-Fi little button. You stick it to your clothes washer, and if you hit it, it just orders more Tide for you.

So the idea is take things to the cloud, have a local proxy for the real-time stuff so that just implements stuff. But the beauty of that is you can now insert the rules in the cloud layer. You can access the cloud using just your phone. I think this will become a very important interface.

You can maybe have Siri interfaces. You can say it through Siri or Cortana. Don't do this; you can change rules. And this is really going to be the future, right? And then you can have this firewall, the cognitive firewall, in the cloud to say, listen, don't just take any old instruction. You need a model. You need my permission to do anything, right? And then that really could lead to a sort of a new way of looking at the Internet of Things.

So my conclusion on architectures is if we plan to connect 50 billion things to the Internet, we better get organized. We can't do to ourselves what we did with IT, where legacy IT sometimes rules our future. The anthropomorphic framework is a good framework. The cloud is a nice design metaphor-- avatars, right? We do it in games. It's an easy design metaphor, and its anthropomorphic, and it also replicates real world.

So the software interfaces look like real-world interfaces. Turn the dishwasher on. In the cloud you say, turn the dishwasher on, except you're talking to the cloud interface. And otherwise we will spend a lot of time undoing what they've done. And frankly, and unfortunately, I think we're going to have some major security breaches because a lot of the implementations are not secure. Thank you.

The Architecture of IoT (Sanjay Sarma): Video 3

I'm now going to give you some examples from my work. This is the last segment. I'll join you again afterwards, at the end of the course, to give you some conclusions and summaries. But this is intended to give you a sense of real stuff.

Let's start with RFID. I spoke about RFID in the previous segment. We did RFID in the late 1990s early 2000s. And without realizing it, RFID complied with many design principles that I've shared with you as a part the Internet of Things.

First of all, we used a cloud before the cloud existed. [INAUDIBLE] use the Internet to put data on the Internet. We used an hourglass architecture, and perhaps it is too hourglass. But we said, RFID tags and RFID readers, those are the two standards that need to be unique.

So any RFID tag can work anywhere in the world, with any EPC reader. We, in fact, went to the point of lobbying certain countries around the world, such as Japan, to open up frequency bands so that a tag that worked in the US could work in Japan, because the supply chain is global. We standardized all the interfaces.

But the one thing we really weren't very good, in retrospect that I look back on, is we did not create a good or an easy design metaphor. The standard sort of over-taught it, all right. It's sort of like web services over-thinking its domain.

Now, mash ups are sort of undoing that in a good way. It's like XML perhaps being too complex. JSON is taking a step back. Sometimes you need to simplify to make things more powerful.

I'll talk a little bit about sensors over the years. Outside of RFID, in my lab, we've done a lot of work in sensors. So, for example, using RFID tags, and sometimes not even with the ID part, we have sensors that can detect cracks in concrete.

We have developed soil moisture sensors. What soil moisture sensors do is you could stick it in a garden and it'll tell you if there's moisture so that your irrigating system can deliver moisture there. It's especially useful, for example, in the Arab countries

We've done a lot of work and scanning. For example, we have a system for scanning buildings with infrared cameras to look for energy leaks. You can see here an energy leak in the frames of this window. This was done in Europe. So we don't want to work in sensors around the world.

But a particular case study I'd like to dive into is the case of the connected car, an area that many people at MIT have actually work with over the years. So a typical car today-- and this is an image that shows all the things that are controlled in a car-- has hundreds of sensors-- ABS, windshield wiper, door sensors, voice communications, emission sensors, and so on.

So the problem we looked at a few years ago was, how do you connect the sensors in your car to you? We went through a very interesting design exercise. This is our historic progression to show you how we ended up where we did, which complies with the principles I shared with you.

So we just stuck a Bluetooth dongle on a regular car. It turns out, most modern cars have something called on-board diagnostics, OBD. It's a little port under your steering wheel. You can see it.

And it's sort of funny. When you take your car in for an emissions check, the emissions people plug into your car and say, hey, how are you doing? And the car says, I'm doing good. And the emissions people print that out, give it to you, and charge you \$30.

So we said, why don't we just plug into it? And we built a Bluetooth connection to a cell phone. Which was great, and we thought we done.

[SNAPS]

But then, what followed was an interesting sort of journey. The problem with that is that what happens when my wife drives my car? Does it pair with her phone? What happens if my kid drives my car and I want to see how fast she's driving? I can't do that.

What happens if I get a new phone, or a new car? I keep pairing them? It's actually not a good metaphor, strange enough. And so we came to the conclusion that the shortest distance between a car and the phone is not a straight line. It is actually through the cloud.

So in a strange way, we concluded exactly what I said to you, which is what you really want to do is take your car and mirror all the sensors as an avatar in the cloud. That's really what you want to do. And then, you want to access stuff through your phone.

So if my wife borrows my car, and later I want to see how's that O2 sensor? Did it still act up? No problem, I could see it. If my kid drives the car, I can make sure she isn't driving too fast.

By the way, this is the model that Tesla follows as well. And so the future of M2M communications, as I call it, machine to machine communications. I joke and say it's not M2M, it is an MCM, machine cloud machine communications.

And you'll have apps on the cloud. And you might have an app that comes and looks at your car's data and says, hey, listen, your tires need a change. In fact, by the way, we've done it. You can do that. You can have insurance apps. Progressive Snapshot is an example of that. You can have toll apps. You can pay your toll from your car. You don't need a toll pass.

And so here's an example that really fundamentally addresses the problem we ran into. Of course, you still need the real-time section. When you hit the brakes, you don't want that to go through the cloud. So you need the reflexive piece in the car, and that's a controller.

But it really sort of, to me, confirmed this emerging vision we had a few years ago, that cloud is the future. And we joke-- and in fact, I've done this. I've created a cloud home. And I joke that someday my cloud car will drive into my cloud home.

And I have an Apple Watch, and so there's a cloud me. And my cloud me could climb into my car and drive into my cloud home. So it's a world of avatars that I personally see. We've developed a whole system around this.

We have the dongles, we have the electronics, we have the cloud infrastructure. We've written several papers about it. We have a lot of work in security and privacy to make sure that, if the driver does not want to be tracked, or who can track their data. They don't even leave a record of their driving. All that, we take care of.

But the good news is that-- I'm just going to go through this quickly-- we generate a lot of data, and then we have apps that look at the data. We have this application programming interface. We have security. So this is sort of like Google Circles.

You can say, these are the things I want to track. These are things I don't want to track and so on. You can turn them on and off, interactively. In some ways, with the system, any car can become as a Zipcar. Any car could become sort of an Uber. So that's the advantage of this.

Now, what we then discovered was that richer data has some really amazing applications. So, for example, you contract your driving, you can see where you consumed a lot of fuel, where you spent a lot of time, where you did a lot of emissions. You can see which gear you drove in. We can see how much traffic you encountered, where, and so on.

But the amazing thing is now, in the recent past, my PhD student, Josh Siegel, has figured out some amazing things. Just by looking at this data or by looking at your cell phone data when you're driving the car, he can tell how quickly the temperature changes of the oil, whether your oil needs changing. Because when your oil gets old, it becomes less viscous. And we can see that it's heating up more rapidly.

So he can see, for example, whether your wheels are out of balance. He can see whether your tires need inflating, just using this big data. And all this is enabled by the fact that we can take this data, the car itself uses the data in real-time, so that's great, but then we reflect some of the data on the cloud.

And from that, we're able to back calculate some amazing things about the car. I actually think this could change the way in which cars, dealerships, and car companies interact with each other-- insurance companies, the whole shebang. And if you've looked at what

Telsa is doing with its car, in many ways, some of these ideas are sort of the same direction they're going in.

So look, it's a great pleasure to end with this. What I'm going to do is I will come back at the end of the class and summarize. But it's been really a great pleasure. And I hope you enjoy the rest of course. Thank you very much.

The Web of Things (Tim Berners-Lee): Video 1

So my name is Tim Berners-Lee, I'm a professor at CSAIR MIT, and I'm going to talk to you about the Internet of Things, or particularly the Web of Things, which is, if you like, the top layer of the Internet of Things. At the moment perhaps the Internet of Things is most famous for the level of hype everybody says is associated with it. This is something which says it's a buzz word.

Everybody's talking about it being this overwhelming wave that is coming. Why should we think about something which really is mainly hype at the moment? Well, I think there are some pretty good reasons for doing that. Yes, it may be a wave that's coming, but there are a lot of reasons for us getting organized, so that when it comes we've got our ducks in a row, and we can manage this data constructively. And for that, we're going to have to do some things now before the advance of the wave.

It's exciting thinking about Internet of Things. In a way, it makes data more interesting but, in a way, when people ask me about the IoT they ask me what it is. I say, well, actually, to an extent it's just data. To an extent, it's just data, to an extent it's not. It's just data in that there's a lot of it. It comes in all kinds of different shapes and sizes. Some of it is very confidential and some of it is very public. All these things apply to data. To a certain extent, if you have a system now which, in an enterprise, which manages all sorts of different data, a lot of little of things you learned from building those systems, building your big data systems, your little data systems, your heterogeneous data systems, you're going to apply to the Internet of Things as well.

But then in other ways, actually, it's not just data. Because the Internet of Things is really these devices, they just can't all be turned into parts of the same computer.

[INAUDIBLE] bottom of the river, it's in space. It's not going to be able to run the same software, it's going to have power requirements, it's going to have the communication requirements which are really hairy, so that communications protocols are not going to all converge towards everything being on the internet.

In a way, 25 years ago, when I was inventing the web, there was-- yeah, there was internet, but all kinds of other networks. There were DECnet, there was IBM Net. Every computer manufacturer had their own internet, or equivalent, network. And the internet was the idea of connecting those all together into one network. But also everybody ended up using ethernet, in fact. All the other things, kind of, went away.

Do we think that is going to happen with the Internet of Things? Well, not really. Because the sorts of wire, or the sorts of wireless you need for connecting to something which is in space, connecting to something which is in your office, or connecting to something which is in the lake outside, they're really different. And they can actually stay different. So we're going to end up with a very, very large variation of different things. So the underlying infrastructure is going to be more heterogeneous than it's been for the internet.

The Web of Things (Tim Berners-Lee): Video 2

Security, to some extent, is then just like the security of a web based system. It is different because the underlying protocols will be different. There's a tendency to just feel we need to connect everything, and when we've connected everything then we'll be able to run the right sort of massively powerful software and everything will work. There's a danger there, of course, that from the security point of view, right now, the only reason that internet security really works for computers is because they get updated.

The only reason that your computer is secure is because you make sure every now and again, when it powers up or morning, it asks you-- or maybe it's automatically in the night-- it downloads the updates, so that when somebody has discovered a flaw in the software you're using that they could use to exploit and break into your computer, then you can have that defense against that attack installed on your computer before they get around to attacking you.

That, in a way, is the rule now for anything which is out there on the open internet. When you look though at some of the things-- if you have a little Arduino in your cellar, which is monitoring the temperature of your wine, for example, then that thing is fun, it's simple to use. It's the sort of thing everybody is getting very excited about in the home automation side of the web of things.

But you know what? That thing does not update itself overnight. So there maybe a rule that we have to either keep that thing on a network very much of itself, not allow anybody from the outside any way of getting on to that network in your cellar. Or it may be that we do have to upgrade it. Instead of having your Arduino, maybe you have a Raspberry Pi, maybe you have something-- a PiZero-- something low which gets to the point that it can download another copy of its operating system overnight and switch over, something which you will then end up being connected to, not just connecting very simply without any security, but connecting to using public cryptography, even if it's a little box like that.

The Web of Things (Tim Berners-Lee): Video 3

The main thing I want to talk to you about, in the web of things, is its diversity. Lots of different aspects, dimensions, of diversity. Data at the moment, is pretty diverse. If you look at just one form of data in your life, like when you download bank statements, and you look at your finances, then you're looking at bank statements, and they all look pretty much similar. If you study the weather, then you might look at weather balloons, and weather balloons data all look pretty much the same.

But then, if you took all the things in the different parts of your IoT world, you realize that you've got all kinds of different things. And some of them are very, very fast, producing huge amounts of data. Like when a driverless car is going around the corner, the flood of information into the senses, is very, very fast. So that's very, very fast. In general, data needs to be dealt with. The weather data, when you look at the weather archives, that's a huge amount of knowledge that mankind has about how the planet has been changing temperature. That's a huge amount of data is not changing so fast. Other things are really small amounts of data, like your life may be governed by the dollar-pound exchange rate, if you trade between. Just one number suddenly is really, really important. So, all kinds of data will be out there on the Internet of Things.

Let's take an analogy. Think about the information that you get about a bag of chips when you pick up a packet. It has on it information when you're trying to pick it for the person who is trying choose which sort of chips they want, the sweet potato chips. But I find when you turn it over, it's really instructive. So when you look at this on the back, there are all sorts of things. There are things meant for different people.

So for example, there is the thing for the general public. I love food. You should buy these chips.

For people who really want to know what they're eating, there's a nutrition fact list. Now, this is very different because this is a controlled vocabulary. You can go to look at anything in the store. And it will have the same nutrition facts.

So when you pull this data in from the nutrition facts panel, that data, which is from this controlled vocabulary, which is government-mandated, you can very easily add up, merge with all the data, from the same vocabulary from different things. So if you take the data from all the things you eat, you add it up, you will have your total intake over all these things. So here we're working in a government-mandated standard, where if you're processing the data by computer, you're really streets ahead.

It's really, really much easier to process this data with computer than anything else. But there's other things down here as well. There's a barcode. Not any interest to you at all, very interesting to the person on the till when you check it out. That is just the identity, which will give it its price. And it's part of a relationship between the person who's selling the chips to the grocery store and the grocery store. Because [INAUDIBLE] doesn't see it.

So these pieces of information are used by different people. And there's some, over here, there are all kinds of little icons. I don't know what all these mean because I haven't bothered to look them up. But if I wanted gluten free, I would have learned exactly what the GF icon means. And I would be going through the whole store trying to see only the things which have got the GF icon on them.

So somewhere on here, there's not only data about the chips, there may be the number printed on the packet by the printer, which says which batch of packets it was when he was printing the packets. So you get the idea that when you pick up a packet of chips, in fact what you've got is a packet of data about the chips. But that packet of data, even though it's just one bag, it contains data from all kinds of different people to all kinds of different people in different languages.

And that is going to be true of the simplest piece of data. If you take, for example, you have a pool and a garden and you have a temperature sensor, it may be if you connect that up to an [INAUDIBLE] and it reaches the level of the incoming wire, let's say, two volts. So that temperature reading has got two volts. Actually, maybe there's a piece of software somewhere which shows, oh, two volts, that means that's going to be 15 degrees Celsius.

Then you've got another piece of software, which is actually running the pool heating which says, well, 15 degrees Celsius, the pool temperature, that's an alarm condition because not only is that a temperature but that's an inappropriate temperature. We need to turn the heating on, getting a little chilly in there. So that you blow out that one temperature sensor that you've got in your pool.

And imagine that happening for every piece of the smart city, of the smart factory, or of the smart supply chain. Clearly, we're going to have huge numbers of different things. And each of those things also is going to have a lot of things, things people attaching information to it from different points of view.

The Web of Things (Tim Berners-Lee): Video 4

One of the reasons why it's important to think about Internet of Things, even though it seems to be something that is away, mounting-- coming, but not here yet-- is that when it comes, we're going to have to link everything together. We're only going to be using the data from all these different thing, unless we understand how they relate. If they all come from different companies, and they all come from different parts of the industry, and they haven't done the business of talking to each other, it will be like in your home when you try to stream music for this thing to that thing, and that will only take this play, and that player will only take that play. And it is disappointing. You lose the functionality. Your home doesn't work because these things, these manufacturers have not agreed to stream music in the same format. And they've done it for a number of reasons.

That's just when you think about streaming music, which is pretty simple, between two things, which is like loud speakers and tape recorders. So, if you imagine all [INAUDIBLE] Internet of Things, what we need to do is to get ahead of the curve. We need to make sure that your company understands all the places where it's going to be using Internet of Things data. And make sure that the people who are supplying it will do it in a way so that your systems can link up, like the data together.

So now is the time to do that. In order to get things standardized, you need to be now. You need to be ahead of the hype curve. Maybe we're at the point where, yeah, there's a lot of hype. And maybe it'll drop off and people will realize people aren't using it. But this gives us a chance to get together, maybe people who are thinking of it, and build standards, pick our partners, the people who are prepared to be open about those standards.

The Web of Things (Tim Berners-Lee): Video 5

And he was just talking about one thing, one bag of chips, one sensor. There are lots and lots of other things out there. And the most important part of linking is going to be between different types of objects.

Why do we need linking? A piece of data by itself is not really interesting at all. If you look at all the sites of the data, the government data sites for example, you've got one about temperature, you've got one about potholes, you've got one about snowfall. If actually looking at the pothole data all the way across America isn't very interesting, but actually if you then compare that with the meteorological data and you can see where the freezes have been, then maybe you can write a program which will predict where the potholes are going to be in a given year.

So it's connecting data from different things, linking it together, which gives you the insight. And maybe with the way you'll be competitive will be that you will be able to link data that other people won't have linked before. So how do we link this data? Well, it's really hard work.

If you pull into a spreadsheet data from one part of the world, or from one part your company, and then you pull into the next spreadsheet data from something else you want to compare, it's often a lot of hard work, because these guys haven't talked to each other. One is using Celsius, the other's using Fahrenheit. One that has done all their accounting with the year end in July and the other's done all their accounting with the year end in December.

You can even compare these things. You try to compare these things, and you end up coming up with some really bizarre and incorrect and dangerous results. So what you have to do, then, is you have to go to a lot of hard work. If you're going to link data, that means, typically, you have to go to the people who are producing the data. And you have to persuade them that you're going to talk the same language.

Maybe you'll be using software. And the software which treats that data has been built by people who've done that work. Maybe they have made sure, like the people who do personal financial software, they've talked to the banks, and the banks now all export OFX format-- Online Financial Exchange. You download your OFX files from different banks. On a good day, they're all compatible. You can just put all that stuff together and bingo, you can see your net worth and how it's changing second by second because of that compatibility between different, because of that interoperability.

The Internet of Things will need a lot of interoperability which isn't there at the moment, so it's getting to the point where manufacturers who are going to be involved in the business have to decide, in a way, which side they're on. Are they going to be building a big silo? Are they going to be pooling all the data into the integration hub? Are they going to be then controlling it and holding you ransom because you've got all your data in the system? Or are they going to be open? Or they going to build new systems which will

interoperate well? Will you be able to get your data back out of their systems and use more than one hub, more than one system all talking to each other?

Hopefully, we'll be able to move towards that, but it's going to take standardization. It involves people agreeing to use the same languages. And agreeing to use the same languages means sitting in a room together and listening to the other guy. And people hate that. It's really hard work. It takes more mental energy from an engineer than anything else. So engineers typically will much prefer to spend a week coding up a new system than a day sitting down talking about how to be compatible with somebody else's system. But, in fact, talking to somebody else about compatibility is going to be very valuable.

The Web of Things (Tim Berners-Lee): Video 6

I've used the terms "internet of things" and "web of things" a bit interchangeably, like people tend to use internet and web. In fact, they're very different. And in fact, I'm going to distinguish between them now.

First of all, the internet and the web-- internet is a set of protocols like TCP/IP which connect all the computers together with the internet. Before that there was a web, there was no websites. There was no HTTP, there was no HTML, there was no URLs. So the web, on top of the internet, is the high-level protocols. It's the things which I actually use as an information space.

So by analogy, what's the web of things? The web of things is the high-level protocols. Imagine the internet of things is then thinking of that as all the connections, all the raw data out there. The work that the World Wide Web Consortium, W3C, is doing on a web of things is to look at the whole thing from a high level, realizing we're going to have a lot of things out there. They are necessarily going to be different.

The first thing we need to do is to build a taxonomy of them. We need to be able understand them. We need to be able to inventory. What do we have in a city? What do we have in this town? What do I have in my house?

So when we do that, we need to have languages for describing at a high level what things we are. They will talk different protocols. We can build a very clever device which will talk to them all by talking different protocols. But one of the things we need to do is to actually catalog how to get to each one. What network is it connected to? Is it on a one wire thing? Is it going to be dumped every now and again when a truck goes through the depot? Is there going to be a little local Wi-Fi link that, for a few minutes, just connects the truck back to its base and dumps all the data about the week? So lots of different ways in which data can come. We need to catalog all those.

So the web of things is about building these languages, building these data formats, so that we can describe all the different systems. It's about looking at all the different things out there. It's about looking at all the adaptors, people building all the different hubs. So the data of the web of things is, if you like, metadata. It's data about data. It allows you to construct a description of everything you have, all the streams you have so you can then build things which will take that description and then be able to construct intelligently general purpose devices and systems on top of that.

The Web of Things (Tim Berners-Lee): Video 7

There's other spectrum which I think is important to talk about. We touched on security. And, of course, when people think about privacy aspect, they think about how they can stop other people breaking in to the system.

But it's only one way of looking at it. In a way, it's more powerful to think about the data which you just want to lock down and you don't want anybody to get at as being at one end of the spectrum. The other end of the spectrum, there's data which is really, really public and you want everybody to get at, like your ads. You spend money pushing out ads. OK. Data about your product is going to be something that you want to be pushing out there.

And in the middle of the spectrum, there's information which is-- it's very personal, and it's about me, but I share it with my friends and family and my doctor and maybe the medical researchers. There's data which is enterprise, which it's about the internal supply chain within our corporation. It's something which is not public at all. You don't want to keep it available to the general public and our competitors, but we do need it to be available very, very easily all throughout the larger company.

So there's the spectrum between the very, very personal and most of the completely public. All of those different types of data will be part of the internet of things. When you think about the intimate data, maybe my heart monitor, which might be telling all you all kinds of things about what I'm up to. The second, the public data, may be open government data that I've been pushing for years, to a certain extent successfully, to get governments to make available, because it's so useful to industry. To be able get at data, like when the trains go, where the potholes are, that sort of thing.

And in between, managing all that are in between, managing to make sure that people get access to the data they need, is important. The idea of concentrating on locking down data, I think, just putting it in a box, is a bit old fashioned, because in fact, the reason you have this data is that people are going to use it.

The reason you have health data, for example, it's, yes, you could use it. I want you as a doctor to be able to get at my health data really easily in the event that I'm suddenly found lying in the street. I don't want people to have to go through lots of difficult hoops.

Maybe I also want-- I decided that I want people to be able to use it for academic research. If they're going to use it for research, then I don't really matter what they do with it as long as they only use it for that research and they don't use it to giggle over me, what sort of poor person I am.

What you use the data for is, in fact, becomes more, more important than whether or not you've actually got it. And so that, in turn, means that we have to build systems which are accountable. If I'm a doctor, if I'm a policeman, if I'm working for the government, I actually have access to data about an individual, I got to have access to it for doing

something particular. Maybe it's the health, maybe it's for antiterrorism, but maybe there's a law somewhere which says that I can use it for one particular reason. Right. So if I sometimes do other things, if I have another hat, sometimes I'm doing surgery, sometimes I'm doing administration, maybe when I'm doing surgery I need access that I did, but when I'm administration, I don't.

So I need to build systems which will be able to let me have access to the data but at the same time allow me to track what data I'm using, and tell the system what sorts of things I'm using it for. So that we can build accountable systems, connections between different sorts of data stream, which we have, like, for example, between my health data and my nutrition data from the bag of chips. Suddenly, allows to be lapsed which is much more powerful.

As those connections start to happen, then the benefit to me of having, for example, a lifestyle app, which can check my nutrition and my fitness and my health data from the hospital, from the clinic, and put that altogether, that's really valuable. When you connect all those things, you can start to really deliver. I think hospitals clearly will be trying to do that to be more efficient with patient health.

I kept very keen about open data. I'm also very keen about personal data. Maybe, it's because I'm a geek. Maybe, it's because I've been doing this kind of on and off for 15 years. But I want to be able to pull in all the data about me, and I want to be able to treat it myself. Yeah, through lots and lots of websites, lots and lots of cloud offerings, which will take my data. But they will get benefit by comparing me with all the other people. Well I want to do is to run something which will look at my data and compare it with all my other data, all the other aspects of my life. And people like me, maybe share it with my friends and family. But something that works for me, because it's when it's my lifestyle, it's my data.

I have a suspicion, I have a theory, that this data about me that is coming from into other thing is I think it's going to be more valuable to me than to anybody else. So I want to see us move away from the fact I don't actually have data, because it's being monetized somewhere else, because I suspect that the data that you're going to get from monetizing, that treats me as just one dot somewhere in some huge data set, when you put me into that insurance company's data set. But to me, well, my life expectancy has a huge, huge [INAUDIBLE] value, and when it makes them make decisions about whether eating chips or not, then that's something which I think is a value that is greater to me than anybody else.

The Web of Things (Tim Berners-Lee): Video 8

So we talked about the diversity. There's one area that we haven't really gone into. A lot of people, when they talk about the Internet of Things, it's all about senses. It's all about the flood of data coming in. In a way, the fear of the Internet of Things seems to be the fear of this deluge.

But just as I've always been pushing for the web not to be a reading only experience, but it should also be a writing and a creative experience for people, well, guess what? On the Internet of Things, of course, it's a two-way thing. So yeah, you may be reading the temperature from the pool in your yard, but also there is the data bit which controls whether or not the boiler is running to heat the pool up.

So of all this data, most of it is going to be read data. Because the way of the world is that you get a lot more data from your house than you have control of it. So you may have the setting of the air conditioning, and you may have the setting of the heating. Derived from that, you may have whether the heating is actually running at the moment. So there are a few things which you control.

When you're building a system, if you're automating a city, what's the point of getting all this data together, managing to link it, managing to analyze it, coming into a conclusion about what we need to do, and not being able to implement that? So if you realize where the congestion is, and realize that actually by phasing the lights slightly differently, than the cars will be able to move smoother at 9 o'clock every morning, well, then, why not do that?

So actually, not just watching the traffic lights, controlling the traffic lights, because of course, it's really important for a smart city. It's no good just watching your wine get too warm in your cellar. You need to turn on the wine cooler.

So, in general, these systems are effective because they have control bits. When they have control bits, obviously, the whole business of security just is at a different level. You can do a certain amount of harm to somebody by looking at the temperature in the house. But you can do a whole level of other harm to somebody by turning on and off the furnace.

But otherwise, it is important to think about these things as read/write. Just as when you have communications between-- up and down the supply chain, for example, you can change data already about prices, to a certain extent. Sometimes it's a feed of here are our prices. Sometimes it's a negotiation. Here's our price. Hey, can you do better? Here's our suggestion. And there's a back and forth.

With lots of things for the Internet of Things, sometimes it's a one-way feed. Sometimes it's a back and forth. Sometimes it's just a feedback. Sometimes you have large systems which produce an understanding. And then the understanding then produces an action in another very different system.

Like we look at all the global warming and we decide that, as we watch the ice melt, and as we realize that it's a gain, that we have record temperatures in October, the result is we decide to reduce drilling, decide to leave the oil in the ground.

So sometimes it's a feedback system which is a very, very large loop involving all kinds of different things between the source and the action. Sometimes, if you're just trying to keep your car between the dotted lines on the road, then it's a very, very fast feedback loop with immediate and very important effect.

The Web of Things (Tim Berners-Lee): Video 9

So the Internet of things is a wave, which is coming, and it is going to be very valuable. And it will be much more valuable to the people and companies who have figured out how to use it in advance, who have made standards, who are prepared to cope with the massive diversity and different sorts of information about a given thing, companies who learned to link together things of very, very different types, and create meaningful information. Companies who've, from this diversity, managed to make a consistent view of what's going on and have then designed things which will react to it in a timely fashion and produce the right actions.

The web of things is about building standards that help a company, help an organization, help a computer, step back, look at everything that's going on, and understand it, and build these complex systems by doing integration across all that diversity, all that complexity, and all those feedback systems at different speeds. It's exciting that in a way we have the opportunity, perhaps a little bit because of this hype cycle, while everybody in the media are talking about the hype, about how exciting it's going to be. Meanwhile, those of us who are building serious systems have a chance to talk to our partners, pick who we're going to discuss, and build systems which will solve these massive issues which we will be faced with of diversity, of the data under linking. So, exciting? Yes, but we have work to do.

Lessons from the Internet (David Clark): Video 1

Hello. My name is David Clark. And I work here at CSAIL at MIT. The topic of this part of The Internet of Things lectures is not about things. It's about the internet. By way of background, I've been working on the design of the internet for about 40 years. So I'm definitely a internet person and not a thing person.

When I first heard the phrase, internet of things, I was mildly amused. Because of course, this is not the first buzzword we've had for this technology. Before we had internet of things, there were sensornets.

And the sensornet community was divided by a terrible debate whether a sensor, or as I'm now going to call it a thing, was to feeble to run the internet protocols. And half the community said, well, of course it's too feeble to run those protocols. I have to go and invent a whole new protocol suite. Whee.

And the other groups, of course you can run the internet protocols on a thing. Well, I heard that before, because in the 1980s somebody said to me, well, the internet protocols for design for a mainframe. And a minicomputer can't possibly run the internet protocol.

So I have to go and invent a whole new protocol suite. How exciting. And then we had the personal computer come along. And I have to say the IBM PC was pretty feeble back then. I wrote the first implementation of the internet protocols for the IBM PC, I would say somewhat in spite.

And they fit. And of course, then you could imagine somebody saying, well, they'll never run on a laptop, a tablet, a smartphone. And the answer is they run just fine on all of those things.

But that masks the real debate. The debate is not whether these devices are too feeble to run the internet protocols. The question is whether the internet architecture, writ broad, is the right framework in which to be hooking up things.

So this topic is concerned with the question of what we can learn from the internet and my experience with the internet about the applicability of the internet way of thinking, not the internet protocols, in the context of how to hook up things. I'm going to do that by looking at a variety of considerations about the internet. I'm going to look at some goals.

The first one is a rather oddly named one. It's the constraints that deconstrain. And I'll explain what that means. Then we're going to look at global reach.

We're going to look at heterogeneous interconnection and generality. And then we're going to look at some areas of the internet architecture where I'd say we didn't perhaps quite get it right. I'm going to look at network management, security, mobility, longevity, and application design. And then I'll wrap up with a few general lessons to close. In the next section, what we're going to look at is the constraints that deconstrain.

Lessons from the Internet (David Clark): Video 2

In this segment, we're going to talk about the constraint that deconstraints. But the way to explain this is to use its alternative title, which is the internet hourglass model. And on the slide I've drawn a picture of the internet. What I've drawn is at the top, a space where there's a whole bunch of applications up here. And down at the bottom, I've drawn a space where there's a whole bunch of technologies. Everybody knows what the applications are. This is the web, email, World of Warcraft, whatever it is you want to do. Tremendous diversity of stuff up there. And down at the bottom you find Ethernet, and Wi-Fi, and fiber-optic links, and satellites, and undersea cables, and so forth like that.

To a user of the internet, this whole picture is the internet. I would call it the internet experience. But to an engineer, the internet is this stuff right here. It's this very simple layer. And the reason we draw it narrow is to imply the range of diversity. You have tremendous diversity up here where you look at the applications. You can have almost any kind of application you want. But this area down here, this is where everybody has to agree. You have to agree on the internet protocol.

Now here's why it's powerful. Any application that can run on top of the internet protocol-- or if that's a little hard, running on top of TCP which sort of makes it a little friendlier-- you can run that on the internet. And similarly, any technology you imagine putting underneath it, as long as it can support the basic idea of moving a stream of bytes from one place to another, we can use that technology to support the internet. So an engineer would say the technology is underneath the internet, the applications are on top of the internet. And what we really care about is just that very simple layer in the middle.

So I draw the middle part narrow to suggest there's less diversity. Why do I call this the constraint that deconstraints? If you commit yourself to that standard in the middle, then I have decoupled what you do below that interface from what you do above the interface. And therefore, I have deconstrained your ability to build applications, I have deconstrained your ability to innovate in the technology space. Now, this idea of the constraint that deconstraints-- and that phrase in particular actually doesn't come from the internet space, it comes from the field of biology, where people study the structure of cells. And they talk about a few critical issues having to do with the way live cells are put together that allows tremendous creativity, if that's the correct word to use, in what cells do in exchange for constraints, mostly having to do with how energy is produced. So there's a rich history and a whole bunch of fields of looking at the critical placement of interfaces, and how the constraint of the interface allows you to deconstraint all the other stuff that goes on around it.

Now, having said that, there are limitations to this architecture. And if I look specifically at the internet, if the technology below the narrow waist has a particular feature-- for example, it has broadcast-- the upper layers could take advantage of that because the internet interface doesn't really give you a very good way to take advantage of local broadcasts. Similarly, if the technology below the narrow waist has some limitations, there's no direct way for the application to know that and to compensate for it. What

happens in the internet today is the application measures how the internet is working, and it doesn't seem to be working very well maybe it adapts to it. But there's no way it can go down and say hey, network, why don't you work differently? Or what's wrong? We've insulated both the innovation and the understanding of faults across that interface.

The wireless community in particular has said, look, wireless is an impaired technology, and we need to be able to adapt the way the application works to the specifics of the particular wireless technology you're using. And the internet doesn't let us do that. So we have to go design a whole new set of protocols.

I'm going to return to this hourglass model later. I'm going to talk about how it applies to IoT. Now that I've got this idea in play, I'm going to go on, and in the next segment, look at a couple more of the goals that I've put for the internet.

Lessons from the Internet (David Clark): Video 3

In this segment, we're going to talk about a couple more of the goals that I identified for the internet. We're going to talk about heterogeneous interconnection, and we're going to talk about global reach.

In fact, heterogeneous interconnection was much more an issue in the beginning than it is now. The first implementation of the internet protocols that I did was on a machine that had 9-bit bytes and a 36-bit word, and if you think about how to deal with a protocol that thought you were going to have 8-bit bytes and a 32-bit word, you could imagine there was a lot of grunge. In some sense, the internet's driven all that out of the system, which has given up, so everybody has 8-bit bytes, and they put them in the same order.

I think that's an interesting question of whether the Internet of Things is going to introduce a new generation of extraneous heterogeneity that we're going to have to deal with. But that's a question to keep your eye on as you go forward.

The second was global reach. The idea of the internet is that any computer was equally likely to want to talk to any other computer on the internet, so everything was globally addressable and globally reachable. That empowered all kinds of applications, but of course, it also empowered an amazing range of security attacks, because any machine in the internet could now send me a malicious packet.

The interesting question about Internet of Things is do they need this degree of generality? Do things need to talk to everything, or are things going to talk to a select set of machines on the internet? Again, I pose that as a question which we will keep track of as we go through this, and you will keep track of as you go through the subsequent topics in this course.

The next segment we're going to look at is generality.

Lessons from the Internet (David Clark): Video 4

In this segment, we're going to talk about generality. Generality was one of the prime objectives of the Internet. We were hooking computers together, and computers are general purpose devices. So the Internet had to be a general purpose network because it had to be able to carry any kind of traffic that the applications wanted to send.

The Internet has absolutely no idea what it's being used for. It just moves bytes. This idea makes a lot of sense to a computer designer. After all, we design computers without knowing what they're for.

This idea makes no sense, for example, to an engineer who designed the telephone system because the telephone network was designed knowing that its purpose was to carry telephone calls and that knowledge is embedded in every level of the telephone system's design. If you look at the early design of the cable television system, the designers knew exactly what it was for. It was to deliver television, and it was optimized for that purpose.

So here's the question about things. Are they general purpose devices, or are they fixed function? In fact, we've had fixed function devices on the Internet for a long time. We have had fixed function devices, like printers, hooked to general purpose devices, like computers. But as we think through the space of things, we should ask-- are there drawbacks as well as advantages to the design of a network that's very general?

Because the Internet is a general platform, it doesn't know what the applications are that are running on top of it. And by the way, I know the term for this is it's the "open" Internet, which means you can do anything on top of it. It's a fantastic platform for innovation. All kinds of applications have been invented to run on top of it. It separates the role of the Internet service provider from the role of the application designer, which was probably a good separation, because it let two industry segments grow up.

On the other hand, because the network is general-- and another word for that is "dumb"-- it cannot help when things go wrong. It cannot help you with management, which is a topic I'm going to talk about a little later. It can't help you set up your application.

It can't figure out if the application's not working right. The Internet can say, well, I've delivered every packet I saw in the last half hour. Of course, I haven't seen any.

The user is sitting there tearing their hair out saying, "What's wrong?" And the Internet's saying, well, I forwarded every packet I got. It certainly can't help with application level security, and there are a bunch of other issues that it can't deal with precisely because it was designed not to know what you're doing, which is a polite way of saying it's a general platform.

Now, how does this map into the Internet of Things? There are two views. The first view is things are miniature general purpose devices. We are carrying around very small general purpose devices with us today. They're called "smart phones."

All right. They can do almost anything. Right.

So maybe a thing is like a general purpose computer, but it's even smaller than a smart phone or maybe some things are big. And they may want to connect into different high level systems. You may have a thing that's feeding into a lot of applications. That's one view.

The other view is a thing is fixed function and fixed purpose. And if I look at the thermostat in my house, I don't think of that as a general purpose computing device.

It has a one and a half jobs. The first job is it controls the temperature in my house. Now, it has another half job, which is maybe it's telling somebody out on the cloud-- another buzzword I hate, but we'll get back to that later. It's telling something in the cloud what the temperature of my house is, and maybe it lets me control it from my smart phone and so forth. But that's still a very fixed function.

If you look at industrial controllers, they have a very specific job. They're monitoring pressure. They're actuating something.

Look at a heart monitor. I don't want that to be a general purpose device. We know exactly what it's supposed to be doing. Another way of thinking about the fixed function is that these devices are being used in a specified context. You might say managing the temperature of my house is a context in which I want certain things to happen.

I'm going to come back to this term of "in a context" throughout this talk. So let me just explain, perhaps by analogy, how to think about this. If you said do you have contexts like this in the Internet, well, the right place to look is look at the application level. The e-mail system is an application context on the Internet. In the email system, we have standards for how you send and receive email, we have standards for how mail senders and receivers talk to each other.

And similarly, in the web, we have standards for how browsers talk to servers and so forth. That's what happens at design time.

At run time, what happens is that a component has to be introduced into the context. And I suspect most of you have gone through the exercise, which is getting easier over time, of enabling a new e-mail client to read your email. It used to be you had to type in some IP addresses and you had to fill in the protocol and the name of your server and so forth.

That's ugly stuff. Humans have to put up with it. And that takes us to the topic of management, which I am going to say is the first of the flaws that we had in the Internet and the topic of the next segment.

Lessons from the Internet (David Clark): Video 5

In this segment, we're going to talk about network management, which is the first of the topics that I identified as a flaw or perhaps an area of incomplete design in the Internet. The term management refers to the way networks are set up, the oversight of the network, diagnosis of faults, and so forth. And the word tends to imply a human component.

We talk in the network about the data plane, which actually moves the bytes. And then we talk about the control plane, which is the automated tools that, for example, make the routing work so that the data plane can function. And when that's not enough and a human gets involved, we call that management.

Management applies to the network itself, people who have to set up and debug the network. And it also applies to the applications and services that are running over the network. In the early days of the Internet, we didn't know how to think about this. We could barely get the bytes from one side to the other.

And moving the data was so hard we didn't really think about management. It was just done by really smart guys sitting in front of the computer and typing at it. But that tradition continues today. If you go to academic conferences, we continue to focus on the data transport.

We focus on performance, we talk about patterns of communication, and we've sort of pushed aside some of these more mundane and frustrating problems like the fact that people tear their hair out trying to set up a whole network. And this does lead to this misguided question that I referred to earlier, which is when people think about Internet of Things, the question is should the data plane of the Internet involve using the Internet protocol and perhaps TCP? It's the wrong question.

Now, overall I would say, as the previous slide would suggest, I think the overall state of Internet management is poor. If you ask how do even network professionals debug the Internet today, they run two amazingly stupid tools, one called Ping and one called Traceroute.

Ping is simply a packet to a machine on the other side of the net saying are you there? And with luck you'll get an answer back. And traceroute is, by the way, what machines did we go through on the way from me to you? Those are our best diagnostic tools. And it's an embarrassment to me that actual users know how to run these things.

I mean, can you imagine if the telephone system required you to debug phone calls by running traceroutes through the telephone switches? Makes no sense whatsoever. If the Internet management is poor, application management is even poorer.

Email works great when it works. But how many times have you sent an email and then something happened to it? Where did it go? Did you get that email? I sent it. I know I sent that email.

Maybe a server is down. Maybe some transfer failed. You have no idea.

In fact, even the people who operate the mail system sometimes have trouble telling that it's not working. When websites fail, what do users do? They tear their hair out. They get frustrated. They go get a cup of coffee.

This is a space that is completely screwed up. I think both users and network operators deal with inexplicable failures and faults. If you have ever set up a home network, if you've ever tried to configure a printer, that's getting a little better. If you've ever tried to configure the security parameters of something, you understand this is just an area we didn't get right.

Now what do we say about things? Computers are somewhat powerful. They have keyboards, they have displays, you can sort of type at them, you can see what they're saying. A lot of things are very simple. They may have a push button on them. They may have an LED.

I'm going to go through later different strategies in the consumer thing space of how to hook a switch to a light bulb. And it turns out it's really hard. In the dumb era, you just ran a wire between them. And not anymore, right?

Now, how can a user diagnose what's wrong if a thing doesn't seem to be working? Let's imagine that my thermostat is no longer telling the cloud what the temperature in my house is. What am I supposed to do? How do I do debug it?

Maybe you try to attach to it using a web browser and see if it has anything to say. I'm going to tell you a secret because you're taking this class. Most people know how to hook to their home or router to see what's wrong. You go to a web browser and then instead of typing in HTTP followed by a name, you type in `HTTP://` and then you type in a number, which is 192.168.1.1. And now you're talking to your home router and maybe it will tell you what's wrong.

You also have something in your house called a home modem. It's either a DSL modem or a cable modem. If you'd like to talk to that, type 192.168.100.1. It's incredibly stupid I'm telling you stuff like this. Nobody should have to know that. But that's what you go through when you're trying to diagnose something.

And I talk to a lot of my friends who are not computer scientists, they're merely people. And I say, how do you debug this stuff? And the answer is, well, I power cycle it and if that doesn't work, I throw it away and buy another one. That is the typical management strategy. And that's how things are going to be managed unless we get this space better under control.

So here's the lesson I want to say about management. The problems of management are poorly dealt with in the architecture of the Internet. And the issues are going to get even more pressing because things are simpler. They're less able to help out when something

goes wrong. So when you think about Internet of Things, don't ask about how the data gets transferred. That's trivial.

Ask about the architecture for setup and diagnosis. And notice when you say Internet of Things, it tells you nothing about these issues. You have to dig deeper. So that's what I'm going to say about management. And the next segment is going to be about Internet security.

Lessons from the Internet (David Clark): Video 6

So, in this segment, we're going to talk about security. And I have to begin by saying that I could talk for five hours about this, but they're not going to let me, so I'm only going to talk about it for three slides. So I have to discuss this very briefly.

It's a very complex topic. It's got many dimensions. It was considered in the early days of the internet, but the issues and the requirements have greatly evolved. And I'm going to give you some dimensions of security that we can use look briefly at how this relates to things. One aspect of security is two machines that intend to communicate with each other that are being attacked while they try to communicate. And the standard solution to this is to try to encrypt the communication. And, in fact, probably in one of the later topics, you will hear about encryption and whether things can do encryption. By the way, that's the next "can it do it?" question. Once you get over the question of whether a thing can run TCP and IP, the next question is, can it run an encryption protocol? And the answer is, probably it can.

Associated with encryption, of course, has to be some key management schemes. Encryption doesn't make any sense unless you and I share a key. And the problem with security in this space is, how did you get the key? How do you know you have the right key? And how did you keep somebody from stealing it?

Another problem is that one endpoint attacks another. My computer's just sitting there minding its own business, or maybe I went to a website and it turns out the website's dodgy, and it tries to download malware, and all kinds of horrible things happen. What do we do about that? Encryption doesn't help there, because I deliberately went to a website, and it's sending me something, and I was expecting to get something. It's just that what it's sending me is malicious.

The solution there, unless you want to restrict the devices you talk to to be ones you've previously decided to trust-- which is to say, completely throw the generality goal out-- is, you have to somehow design the system and the application, the end node, the software, to protect you from those harms. Which is to say, we're going to let you do risky things, but we're going to try to make it safe.

The other problem you have is, when your computer is successfully attacked, somebody's found a vulnerability, you have to be able to upgrade your computer to put in a patch to close the vulnerability. And that's going to raise some interesting questions about things. Can they be upgraded in the field?

The third security problem, which I'm not going to talk about a lot here, is that one part of the network attacks another. That's really outside the scope of "internet of things," but I would point out this does happen today on the internet. And the solution is the better design of technical mechanisms, not to achieve technical optimality, but to tolerate the presence within the system of untrustworthy actors.

Let me go back to the word "context" that I used earlier in talking about how to position things. I used the word "context" to describe the environment in which a thing or an application exists. And an important aspect of defining a context is its security parameters. And, again, by example, if you say, well, what is the security context of the web? Well, it actually comes from a series of machines which are deployed around the internet called certificate authorities, which are part of the certificate hierarchy system. I note in passing that this system is flawed and is a source of many insecurities. But the point is, when you as a browser attach to a server, the reason you know who the server is is that the security architecture implied by the certificate system is telling you that you've gone to the right place.

If you look at email, I would say, overall, its security context is poor. We have talked about signed email. Of course, if you want to do signed email, then you have to decide how you trust the signature. But that would be the solution if you tried to make email more secure. If you look at a centralized application like Facebook, in some sense, they have defined the context, and, in passing, have defined the security context. So, if you're trying to understand what is secure or not about Facebook, or you're trying to understand something about the privacy policy or not of Facebook, you essentially go to the service as it was created by its owner and they have centralized all of that.

Now, what is the security context in which a thing will exist, and where does that context come from? As I said earlier, another wrong question to ask is, "can a thing encrypt its data?" Because that's not the right question. The question is, how is the security context set up? When you first put a thing in, how do you help it know what machines it's supposed to talk to, what their identity is? How do you keep it from being attacked?

And this brings us back to the issue I was talking about earlier, which is management. Security management-- in particular, the setup of the security parameters-- is a major dimension of the problem that I was calling "management," and that ties together security and management. Now, let's talk about security of things. It's possible the problem is simpler. Why do I say that? Well, maybe things don't ever have to talk to nodes which might be untrustworthy. Maybe, because things are fixed function, all they do is talk to a known set of machines, and we can sort go outside the technical components of the system and say yeah, by the way, I trust that server. I have good reasons to believe that server's trustworthy. So I don't worry about being attacked by the server I'm trying to attach to.

Perhaps we can hide things behind a protection interface, like a firewall. Firewalls prevent a wide class of attacks against your computer. In particular, firewalls mean that the machines behind the protection interface don't have IP addresses. It's hard to send malicious traffic to a machine if you don't know its address. It has to reach out and touch you. Of course, once it does that, it's fair game, and you can come back and attack it.

So, here's the question. When you are designing a thing, what presumptions are you going to make about the context in which the thing is used? Shall you assume that it's always behind a firewall and you don't have to worry about certain kinds of attacks? We

don't know how the context for things is going to emerge, but, if a thing is fixed function, then perhaps its security context can be fixed. And that's good. But what's bad is if you ever have to type an encryption key into a device without a keyboard. If you ever have to set up a device by putting an encryption key into it, you're doomed.

And the final point I want to stress, and I can't stress this enough, although I'm only going to touch on it this one place-- every device on the internet that has ever been shipped has had a security vulnerability in it. And every device that's on the internet has either lived with that vulnerability until the end of time or we have patched it. Are we going to design things so they can be patched, or are we going to discover that things live with their vulnerabilities till the end of time? And that's critical, because we have to ask, how long do you think things are going to last? And some things may last a very long time, as I will get to later.

So, here's my lesson about security. Saying that a thing can encrypt its data is not enough to make a system of things secure. You need a security context or you need a security architecture. And think about a designer trying to bring a thing to market. Where does that context come from? Where does that framework come from? A lot of consumer things today have to solve this by designing their own context as part of the product, because there is no context in which they can live.

So, I have a thermostat in my house. Turns out it's not a Nest, but I mentioned a Nest here because it's a very famous one. To bring that product to market, not only do you have to ship the device, but there are web services that support this thing. Think about the Amazon Echo, which is this little device which you can speak to, and you can say, hey, Amazon, what's the temperature outside? That only works because it goes out on the web. And so the entire web infrastructure is sitting behind that thing to make that a viable product.

But it's actually a tremendous barrier to the introduction of things if everything involves the creation of a whole new internet level context. Think about light controllers. How do you cause your switch to turn on your light? If you think about industrial control as opposed to, say, consumer devices, those things are often designed to fit into a pre-specified context. But it's often a closed architecture. It comes from a vendor, and the vendor says, well, buy into my world, and then each of your things will just fit in. But that's the opposite of the internet open architecture. That is an architecture which, in exchange for convenience, and perhaps an easier to understand security, is a closed system.

So, what the internet teaches us in this space is the power and the cost of an open system. If there's any guidance about security, it's not going to come from saying internet of things. You're going to learn about security by looking at applications, not the internet. The next segment is mobility, which is the next topic in which I'm saying maybe in the internet we didn't get it quite right.

Lessons from the Internet (David Clark): Video 7

In this segment, we're going to talk about mobility. In fact, mobility was part of the early internet, but we didn't understand all the issues. What we did was integrate a wireless network. It was a packet radio network. But we did not fully deal with mobility.

Here's a key issue. How does the device maintain its identity and its reachability when it moves? In the internet, what we did is we lumped together those two ideas, and we associated them with the address of the device. Every device on the internet has an IP address, which in its current form, is a 32-bit number. But what that means is when a device moves, it gets a new address, so it has to reestablish its identity.

That was not a good design. We should have separated identity from location. Now, some things will be very mobile. On the other hand, they may be fixed relative to their path into the larger internet. Let me give you an example.

There are a lot of things you carry with you, like your, let's say, Bluetooth microphone, which get to the internet by going through your smartphone. So your Bluetooth microphone in a physical sense is very mobile, but it thinks it's fixed, because it's always talking to your smartphone. So maybe that's wireless, but it's not really mobile.

There are going to be other things that never move. There are going to be some stressed sensors that are poured into concrete and left there for the life of the bridge. They ain't going to move. So there's going to be a highly variable manifestation of mobility in thing space, and we don't yet know quite how to think about that.

I want to talk about wireless a little bit. Wireless connectivity is a basic challenge to the design of the internet. First of all, it provides modes of communication, most obviously broadcast, that internet protocols cannot exploit. And broadcast is a very powerful tool. One thing it does is it helps with the management problems associated with setup. A machine can send a message out saying, anybody out there prepared to help me?

Secondly, the strict layering of the internet, as I described earlier with this hourglass waste, prevents cross-layer adaptation that can greatly help to mitigate wireless impairments. Wireless is an incredibly variable communication space. And the potential variation is a challenge to some of the core protocols of the internet-- in particular, our congestion control protocols and our error recovery protocols.

Now, you could say, if this is so bad, why do people use wireless to connect to the internet? And the answer is, it works well enough. And it is so compelling as a service that you'll put up with the fact that sometimes, your smartphone is not working right. We just accept that as the price of this convenience. You have to get to the internet.

But the internet of things space may be different for a number of reasons. Of course, one of them is there isn't a human who says, well, I'll just walk over to the next room and see

why this stuff works better. But there's a particular aspect of wireless that I want to stress with respect to internet of things, and that has to do with industry structure.

Wireless access is not just a technical issue. And there's another lesson from the internet, which I bring up here, but is going to show up all sorts of ways through this analogy to the internet that we're talking about here, which is technical standards induce industry structure. And I said earlier about the hourglass, that that cut point defined by the internet protocols separates the people below who are the internet service providers from the people who sit above, who are the application providers. Internet service providers, in general, are not in the business of selling applications, or if they are, it's a separate business.

So the industry structure, in return, defines who controls the shape of a context. In wireless, I can give you two obvious examples. If you look at Wi-Fi, Wi-Fi is typically deployed by individual consumers. There is what you would call edge empowerment. It's highly decentralized. Anybody can go buy a wireless home router and put it in their house.

Of course, this implies edge management. You have to set the foolish thing up. And if it doesn't work right, the problem is yours. It's nobody else.

Contrast this with the cellular industry. The cellular people say it is a centralized system. We control it. We will make it easy for you to use. But by the way, we own you. The cellular industry would love a world in which everything you buy comes with a monthly service contract in order that it connects to the internet. It isn't going to play out that way.

But if your thing is not attaching to the cellular network and coming complete with a monthly service contract, is it connecting to Wi-Fi, which means it only works in your home? What is the industry structure that's going to provide the communication infrastructure for inexpensive things, many of which don't have much to say? Video cameras have a lot to say, but temperature sensors actually don't have a lot to say. I'm not going to buy a monthly service contract so there's a device that periodically tells me it's still 72 degrees. That's ridiculous.

So now you might say, well, what can things do to control the contexts that are emerging here? Well, think about your smartphone. Your smartphone empowers you. And the reason it empowers you is it probably has at least three different radios in it. It probably has a cellular radio. It probably has a Wi-Fi radio, and it probably has a Bluetooth radio.

And the control over which of those radios is used is in the device. It's not under the control of the cellular provider. It's not under the control of the Wi-Fi system. So the fact that there are multiple radios transfers control to the owner. Are things going to have multiple radios in them, or are they so feeble, as I said earlier, that they can only have one radio? That may actually lead to an industry structure in which somebody manages to capture more control over the context than you would like.

I guess the next question about things, of course, is how mobile does each thing really need to be. So let's do lessons about mobility. There's a point that's going to come up several times. I brought it up here. Different things will have very different requirements. They're going to have different capabilities, and they're going to be different objectives.

There is no single internet of things. They're going to be smart things and dumb things. They're going to be mobile things and fix things. There's going to be variation from consumer space to industrial control, perhaps to medical devices, to whatever else you can think of. And these are going to lead to different contexts. And what that really means is different internet of things.

So the word-- or the phrase-- internet of things is a little deceptive. And yeah, it's great as a buzzword. We all know what we're talking about. But that phrase doesn't really tell us what's going on here.

The modes of connectivity are just one example of this diversity. When you're looking at what seems to be a technical issue, look beyond the technology and look at the larger context. Look at industry structure in particular, because that's going to tell you who controls the context.

We'll go on to the next segment, which is about longevity.

Lessons from the Internet (David Clark): Video 8

So this segment is about longevity. And you might say, why did I list this as a flaw on the internet? Because with respect to longevity, the internet's been an amazing success. It's been around since the 1980s. The issue is at the higher levels. And I'm going to focus specifically on names.

In the internet, we have addresses for things. And then we have names. They're called domain names. And the translation from a domain name to a machine is done by the Domain Name System. And this is a way to separate the name of a machine from its location.

But those names were designed to name computers. And that's not really what we want to name. People don't want to attach to computers today. They want to attach to a service. Or they want to retrieve a piece of information.

The designers of the web made a pragmatic decision which I think in some respect haunts us, which is they embedded a domain name inside a URL, which means that when you move a piece of information from one machine to another machine, its name changes. So its name is now tied to its location, which really interferes with our ability to name information over the long run.

What do we actually want to name? As I said before, you want to name information. And the architectural challenge, which we ducked completely, and which to a certain extent the web ducked, is how do you find information as it moves? If the name tells you where it is, then it's easy to find it. Until you move it. And then you can't find it anymore.

If the name does not tell you where it is, then there has to be something else attached to it that says, oh, and by the way, it's probably over there. So why don't you go look? We didn't solve that problem.

The next thing that users really want to attach to is they want to attach to services. They live at a higher level. They live up in the broad part of that hourglass where I had applications. They want to go to Facebook, or Twitter, or email, or World of Warcraft, or whatever they want to do. Users want to reach a service. And they don't care where it is.

So some part of the system has to map from a service to a location. Now, we start with the Domain Name System. And the obvious question is how do you find the Domain Name System. Well, we have to do a little magic trick there. Right?

We build into a lot of machines a magic list of all of the IP addresses where the domain name servers are used. And when you want to find a domain name server, you look at that magic-- the software, not you-- looks at that list, picks the IP address, and goes and looks. And we have some special addressing modes in the internet to make it easier to find the domain name servers.

Now, once you've got that, a lot of higher level services depend on the DNS to map a service to a location. If you look up a very popular service like Google, the Domain Name System may give you back a different address given where you are in the world when you go to attach to it. So the Domain Name System is actually taking the job of mapping services to locations.

Now, is this a good idea? Is it an afterthought? It's something we didn't engineer. It just happened. And now that it's happened, of course, we have to make it work. Because everybody's depending on it.

So now let's go to internet of things and ask about longevity. What sort of elements in the space of internet of things need to have continuity? Which is to say, they need to have long-lived names. Do physical objects need to have long-lived names? Specific things like the thermostat that I bought to control the temperature in my house.

Maybe the physical thing does not need to have a name. I replace the thing periodically. I need to name its role. It's the second floor thermostat. And any physical device could be playing that role. I can take one device out and put another one in. Maybe I need to name the data from sensors, the temperature of my house at 2:30 yesterday afternoon.

Is that an immutable name? Do we need that to last forever? Or is that a transient piece of information? Maybe I need to name actions, like turn on furnace. We haven't really sorted out in the space of things what the elements are that need to have long-lived names.

What we do know is that there are going to be components of this system that last a long time. I think some physical devices are going to last a very long time. Today it's typical to replace your smartphone every two years. You replace a computer every two or three years. There are going to be some things that are probably going to last a decade. OK?

Compare the useful life of a computer to the life of your car. Are you going to have to replace all the electronics in your car every three years to keep its thing architecture up to date even though your car probably lasts 10 or 15 years these days? How often do you replace your furnace? Because we're about to have smart furnaces, right?

How often do you replace your door lock? If you discover that your door lock has a vulnerability and anybody can break into your house, are we going to do a security patch to your door lock? Or do you just throw it away and buy a new one? That's going to irritate a lot of people.

How often do you upgrade your house? Houses last a very long time. And now we're building smart houses. We're going to embed in the smart house stuff which, in the mind of the designer, probably has a lifetime that's the same as a computer. That's an incompatible set of decisions that we haven't begun to work through.

There's one possibility, which is computerization is going to make everything in our lives disposable. But I don't really think that's right. But the alternative is that everything in our

life has to be upgradable. And people say, oh, my thing is too feeble to be upgraded in the field. It's a fixed function device. And besides which, it's buried in concrete. So how can I get to it?

A serious set of issues there. We're going to have to work those through. And we haven't really thought about the implications of longevity in the space of internet things. The next segment I want to talk about, and the last of the elements that I listed as a flaw in internet, is design at the application level.

Lessons from the Internet (David Clark): Video 9

This segment is the last of the ones that I described as flaws. This segment is about application design. And it's perhaps not quite fair to call it a flaw. What I would say is that in the early days of the internet we simply completely ignored how you design an application.

At the time that seemed quite justified. We had no control over what application designers were going to do. They could build any stupid app they wanted. And given that we had no control, why should we think about how apps are designed?

But we started out with an over simple mental model of how an application was going to work. Which is that there was one machine that talked to another machine and the packets flowed from here to here. And indeed the first application we built, which was remote log in, worked exactly that way. There was only two machines involved. The one next to the human, and the machine they were logging into.

But that simple model very quickly fell apart. And in fact the next application we built, which was e-mail, that application has servers inside it. If I want to send email to you, I don't send it from my computer to your computer. I send it from my computer to a server that works for me, who sends it to a server that works for you. And then later you pick it up on your computer. There are actually two servers there.

And you might say, why did we do that? Well, in the early days there was a very simple reason we did this. The internet was so flaky that we couldn't keep the computers up. And the probability that my computer was up at the same time yours was low enough that I actually could never send email to you.

So we said, well we'll put these machines in the middle, and they only do one thing. And we tried very hard to keep them all up. So I sent the email to that machine, and with luck it stays up, and then some time later your machine comes up and you can pull the email down. It was a very simple solution to a problem of robustness in a system that was just barely working.

But we realized, but perhaps not as much as we should have, is that real applications are going to have services, which is to say servers, in them. Similarly when the web was first conceived the simple idea was, well, there'll be a web server and they'll be a client, which we call a browser. And the browser connects to the server, which is two parts.

And as the web has grown up, it has come down with all kinds of auxiliary services. I mentioned the security services, which are the certificate authorities. And of course there are content distribution services called content delivery networks, CDNs.

There's a whole suite of services, which is to say servers, that have crept into the design of applications on the internet. And that's what we didn't really focus on. Which is that

real applications exploit services, which means they are exploiting servers on which these services run-- which are positioned somewhere in the network.

Now, a lot of those services as I described them are supporting services. Some services define the application. Facebook is defined by the service offered by the company, which gives you Facebook. Twitter is defined by the service that the company that gives you Twitter has defined.

But in the context of the web, for example, you get supporting services like the certificate authorities. Those are conceptually quite distributed. Some services are conceptually centralized. Obviously the ones that, in some sense, define the service, like Facebook or Twitter, they are conceptually centralized.

Mail servers, in contrast, are conceptually, and in fact, very decentralized. Content delivery networks by definition are decentralized, because the idea is they're trying to get the content replicated are all around the place to be near you. The domain name system is highly decentralized. So you get this range of designs for services, from centralized to distributed. And that applies both to the physical topology of it, and also to the control architecture.

Again, we come back to this question of industry structure. The technical design of an application, which is to say the set of services that make up the application, define the industry structure. So if you have a centralized application, like Facebook, well of course that emerges as a single-firm innovation. A company invented the idea of Facebook. A company invented the idea of Twitter, or Instagram, or whatever you want to have.

Where do decentralized applications come from? Decentralized applications tend to emerge from design communities with less focus on commercialization and more of a focus on meeting some user level need. I would go back to the web. If you look at the inventor of the web, Tim Berners-Lee, he's very clear he was not thinking about commercialization at the time he invented the web. He was thinking about an open architecture-- and I use the word 'open' in the same sense I talk about the internet-- for information dissemination.

Email was designed as a system for everybody to communicate. And in fact it wasn't thought of as a system where people would make money. Anybody who ran the internet just brought up a little e-mail server, and we just sort of did it.

That's in conflict with the current model of how to make money from the web. Which is invent an app, have control of it, and make everybody come here, and then make money off it somehow. You subscribe to it or you sell advertising or whatever it is. The day I start seeing advertising in my email-- well, never mind. I won't go there.

Supporting services often end up with a decentralized design, because, in fact, it is necessary for multiple people to have control over part of the system. You can't imagine a

centralized domain name system because different parts of the ecosystem are controlled of their own names.

And there's some services that have triggered a highly contested dispute. Such as, should we have an identity system on the internet? And one answer is, well, we'll figure out a small set of actors, or one actor, and they'll just give everybody in the world an identity and then we'll all be done. For social reasons that triggers tremendous resistance, both because we don't trust that actor. We'd be giving them much-- too much power. And as well it means that you can't have control of your own identity.

An alternative design for an identity system would be highly decentralized where I could create my identity. And I could store information about it in a server of my choice. So there might be a competitive landscape of identity servers. That's a contested space. And in fact it's not clear where the resolution of that would emerge.

Remember I said earlier context. Internet of things-- devices-- will be deployed in a context. And a context is really the same thing as application level architecture. So what I'm tying together when I say that we didn't pay much attention to application design in the early days of the internet was we really didn't think about where these contexts come from. But it is the architecture, or the context-- whichever word you want me to use-- that will define issues such as management, security, and so on.

And again, here I am focusing on a non-technical issue. I am focusing on who controls the context. Things could report to a centralized master. If things need to talk to each other, well maybe that's the social network of things. And it's sort of a Facebook of things. And there's some sort of service out there. I'm not saying it would literally be Facebook, but there's some sort of centralized service out there, and all the things talk to it. And then all the connections between the things, and which things are friends of which things are all administered through that centralized service. I don't actually like that vision.

A simple context might be at the other extreme, which is how my light switch controls my light. And I'm going to go through some alternatives there as an example. The extreme, of course, is that the entire control of the context belongs to the individual user, which has the plus that the user is in control. And has the negative that the user is in control, and has to manage the whole thing.

So let's look at consumer internet of things for a minute. And I want to look at the smart home. And go through this exercise that I've hinted at a couple times, which is, how does my light switch talk to my light bulb? This sounds like a joke. Right? How many internet of things people does it take to screw in a light bulb?

But so how do you connect a switch to a light bulb? Option 1 is that you introduce the light bulb to the light switch directly. You could imagine sort of physically getting them next to each other and saying, light bulb, meet switch. Switch meet light bulb.

That's OK until you imagine the light bulb might be 25 feet up in your ceiling, and you had to replace the switch. You going to climb up on a ladder and say, switch me light bulb. Light bulb meet switch. Not going to do it that way. Right? You got to do something else.

And by the way, that doesn't scale. Imagine setting up a building with 1,000 light bulbs and 1,000 switches. And you say, what are you going to do?

What I do in my house is I have a smart controller. I have a little piece of software run on a computer. And so the way this problem is solved is when I buy a new switch, I go to the computer and I manually type in to the computer the serial numbers on the switch. And then I set it aside. And I manually type into the computer the serial number on the bulb, and then I set it aside. And then I go into my computer and I make connections. You know this switch goes to that bulb. This switch, that bulb. This switch, those three bulbs. You can set it up.

Now there's some real advantages to that scheme. The first one is the things-- that is to say the switch and the bulb-- are not on the internet. OK. They can run local protocols in my house. It's protected. The control is local.

There's a word I said earlier. I don't particularly like the cloud buzzword. They talk about fog. Fog is when the little bit of the cloud comes to you. Now I hate that word, because to me fog is something you wander around in and you get lost, and you hit a rock. And so fog is not a positive word for me. But nonetheless in the buzzy space of cloud they've now got this idea that a piece of the cloud is going to come to you. But the consumer has to set up and administer the smart home.

Well, there's an option 3, which is some third party could administer the server in your home. It could be physically in your home, but you outsource its management to somebody else. That's a perfectly reasonable idea, but today there is no industry model to try to make that happen. It's an example of a highly decentralized outcome that could emerge, but only if some set of people design it.

The fourth option, of course, is the control function is in the cloud. The way my light switch talks to my light bulb is it sends a message out on the internet to a central controller. I don't have to administer it now, because it's being run by professionals. And that sends a message back to my light bulb.

Of course what that means is that if my house is ever disconnected from the internet, my switches can't control my bulbs. And you may say, well, I could always go around and just manually turn them on if that's what's happened. And I understand that. But we've already seen episodes where this kind of thing has happened and people didn't understand how thoroughly the internet was interposed until they actually had a problem.

On 9/11, which was not intentionally attacks on the infrastructure, but did a lot of damage to the infrastructure because buildings collapsed on switching centers, and a lot of

connections in New York City were severed. There was a hospital where the doctors were looking at patient records using a portable device. The patient records were in the hospital, the doctor was in the hospital, but all of a sudden the doctors lost the connectivity to the patient record, because the device they were carrying went out into the cellular network.

That cellular network came around through the internet and back into the hospital back end. And that connection was severed by a crushed building. So even though the data was local, and the device was local, the doctors couldn't see the patient records. Not a mistake that is-- that you want to make. Don't do that.

So here's the lesson for application design. The development of a internet of things, as we're using the term, is going to depend on a specification that is at a higher level than the specification of the internet. The specification of the internet is how you put a packet in one side and it comes out the other side. That's not the issue. There may be an hourglass in this architecture-- a point of common connection, if you will. But it may be at a higher level. It may be at a level of a service definition-- or names for things. And, like internet applications, there are going to be more than one.

So this ends the discussion that I wanted to have about flaws or limitations of the internet design as they relate to internet of things. Identified five of them. I talked about management, security, mobility, longevity, and application design. The next segment, which will end this topic, is a couple of slides on general lessons.

Lessons from the Internet (David Clark): Video 10

So in this segment, which is the last segment of the topic, I'm going to close with some general lessons. The first one is that things seem to be physical objects. When you think about them the whole idea of thing is that there's this physical object that you're dealing with. But their real character is not going to emerge by studying their physical character. It's going to emerge by studying the context in which they are intended to operate.

And that raises a very fundamental question. Is the hourglass model the right way to conceptualize how things will work? And I say that because a thing has a physical embodiment, and it has an embodiment at the application layer which is what it's doing, and those two seem to be fastened together. But the whole point of the hourglass was to decouple those two things. I think some later topics will refer to the hourglass model. And as you hear that you should think about whether the hourglass model actually captures how a thing is configured and will fit into the larger context in which it works.

Applications define the way the generality of the internet is restricted to a specific purpose. The internet can carry bytes for almost any app you can imagine. It is Facebook that creates the Facebook experience for you. And it is the constraints and design elements in Facebook that tell you how it's going to work. Applications constrain the generality toward a particular purpose.

And just as there are many applications on the internet there are going to be multiple contexts for things. There will be more than one internet of things. And it is the specificity of the context that actually makes the things interesting, rather than the statement that there is an infrastructure layer, perhaps based on some hourglass conception, that's general. Because that's not what actually matters.

A point I've made several times, and it should always be in mind, is technical design induces industry structure. And you may think you're designing a technical artifact but in fact industry will come in. And it is the interfaces in that system that will help modularize the industry. Just as with the internet, the internet service providers are separated by the specification of the internet from the application designers above it.

So here is a very important, non-technical question to think about when you look at an architecture for internet of things. Do all the business entities defined by the architecture have the incentive to carry out the role that the architecture assigns to them? An example will help because that sounds a little abstract. Internet service providers. The big guys. Comcast, Verizon, AT&T, all these guys, they move bytes. They didn't used to. If you were a telephone company you used to provide a service. You provided telephone calls. If you were a cable company you used to provide a service. You delivered cable TV. Now you're being told your business is just deliver bytes.

And there are obvious questions. How do I make money doing that? That's a commodity business. Why can't I be in the service business? They would actually like to move, in the internet language, up the stack. Go through that internet layer and start to offer their own

applications because they think you could make money there. Then the risk is they will favor their applications over third-party applications.

And that leads to a debate, which is raging in this country. The FCC is now making its third try to impose what's been called network neutrality regulations, which basically says, no guys. You're in the business of delivering bytes. That's what you do. And you're not in the business of favoring certain bytes over others or favoring your bytes over my bytes. You are delivering bytes. That's network neutrality.

When you look at an architecture for internet of things, ask the same question. Why is it that you think the modularity you've induced will actually suit the industrial structure that's being created? And will people fit within your architecture or will they fight it? The power of the internet was its open and general character. And however much we want to think about limitations of that in the context of internet of things, that was what made the internet exciting. New devices could be added at will. New applications could be invented by anyone.

But open architectures do not emerge naturally. In fact powerful players will like closed architectures because it gives them more control, more opportunities to make money. So you might say, well, why is the internet open? Well it was designed by a motivated group and a group that was funded-- funded by the federal government in this case-- who were not seeking financial advantage.

So will architectures for things be open? How open should they be? Would we rather accept a closed space? Because maybe it's easier to manage, it's more secure. On the other hand, you may have given too much power to an industry actor who will then turn around and regulate it in ways that limits innovation, limits third party creativity.

I think that only with a suitable architecture and obviously, given my internet history-- and I'm now going back to the fact I'm an internet guy-- a suitable architecture to me has an important element of openness. Only with that are you going to get a rich base of things coming to the market.

And the final point I'll make is do not underestimate the demand for generality. I've been talking about things as fixed function devices. But people are going to come along and say, can I use that thing in a way that you didn't think of? I know it's a thermostat but could I use it like this? Well maybe the answer is, well, you can't use the thermostat like that. But you could add a new function to the thermostat's context so that you could do something else.

But successful devices will be used in ways that the inventor didn't think of. And an excessive dependence on fixed functionality may, in fact, limit the space. Just as too much generality may limit it as well. That is the final lesson I'll give you. With that I'm wrapping up.

What I've tried to do here is give you some context, some thoughts that you can bear in mind as you listen to the subsequent topics here. These are a lens through which you can look at some of the talks are going to hear as you hear about security, as you hear about this, you hear about that. Look at it using some of these tools. Openness, the hourglass model, management, where does context come from. I hope these are useful tools for you in the subsequent topics you're going to hear. And with that, thank you very much. And on to the next topic.

RFID Deep Dive (Sanjay Sarma): Video 1

Hi, I'm back. I hope you've enjoyed the sessions after the introduction, and now I'm going to take a somewhat deeper dive into some of the other stuff that's happening in my world, but especially into RFID, to give you a taste of how a fundamental IoT technology sort of plays out. And then we'll go into conclusions and we'll wrap this course up. So with that, let's jump right into it.

Maybe just to get going, I'm going to give you a couple examples from my work, for what it's worth. Some of them have resulted in companies. All of them have resulted in papers. Some have resulted in adopted technologies and so on.

One project that I'm very excited about is something called City Lights, and here's the basic idea. What we do is we create a sensor platform which consists basically of a camera and some spectrometers. In the future, it might just be your cell phone facing upwards, and we put it on city utility vehicles, for example, a police car or a utility vehicle that fixes roads. And as it drives around in the night, what we do is we collect information about how much light is reaching the level of the sensor, which is close to the road, from street lamps. And we use that to map out street light intensity in a city. We've done experiments in Spain. We've done experiments in Cambridge. We've done experiments in the UK. It's sponsored by a company called Ferrovial.

And we use that to figure out, for example, where the street is underlit. In some places, for example, you may have commercial lighting until, let's say, 9:00 PM, and there's no real reason for the street lamp to be as bright as it is. Now, as street lamps go to LEDs, there are all sorts of new opportunities to modulate the lighting and to save power. But also to provide safety. So this is a project that enables IoT.

Here's another project. This is something that we started some 10 years ago. What we do is we use long-wave infrared cameras to scan buildings, sort of like how Google Street View scans buildings in the day. We do it by night, but using long-wave infrared cameras.

Now, the thing about long-wave infrared is it detects heat leaks in buildings. So if you look at the picture, you will see several things. First, you'll see an image of a building. And if you look at this building, you can see, for example, that this chimney area is very bright. You can see that these windows are bright, which may mean that the windows are leaky. Heat is coming out.

To enable this, we've created instrumentation systems that consist of a long-wave infrared camera, a regular camera, and near infrared camera, but actually a lot more, which I'll talk about in a second. We put them on vehicles.

It's actually a startup company Essess.com. These vehicles have GPS. They have lidar, like which I show here. And with that, we map entire cities in the night. And it's also, in some ways, a nighttime navigation technology, which will play a part in Internet of

Things. For example, when self-driving cars have to navigate the streets at night, this may play an important part in that. So that's another example of technology that we have developed.

And we actually collect terabytes of data every night doing this.

Here's another sort of a very interesting IoT related project.

There are now technologies that use paper to wick liquids. As you know, paper will absorb liquids. And you can create highly designed sheets of paper. And what they've used here is that paper which comes from a technology that came out of Harvard University.

So the idea is there's this little sheet of paper. Imagine a clinic somewhere in the developing world. Patients walk in. You prick their finger. You put the blood on the paper. The paper wicks the blood. It goes to a couple of different spots where a couple of different chemical reactions occur, with which we can detect whether anemia occurred.

But how is this Internet of Things? What we do is we also attach an RFID tag to this. And the RFID tag changes its behavior based on the chemical reaction that the blood had.

The idea here then is that if you have thousands of patients walk through, you wick, you take blood samples, and you toss it into a biosafe container. And then afterwards, you take an RFID reader and you read. And you can remotely tell that patient number 17 and patient number 42 had anemia. And so it is an example of how we can use the Internet of Things in very different ways.

And finally, I'm going to talk about drones. You can never go wrong with a drone in a presentation. Drones are actually going to play an important part in the Internet of Things. Why? Because they're eyes in the sky. They're sensor packages.

And actually a very interesting different way to look at, drones can provide local connectivity. So let's say that you have sensors in a forest area. And you've dropped a bunch of sensors, and they're looking at a fire hazard. Now, these inexpensive throwaway sensors can't communicate, necessarily, all the way back to the city or to the station. But you can hover a drone over these sensors, or even a helicopter, and the drone can pick up data from these sensors, and then transmit it back long haul. So drones will actually play an important part.

Now, this happens to be a company that was founded by a set of students from MIT. One of them is my PhD student Long Phan. It's a company Top Flight Technologies. And what they do is they've created drones which are hybrid and they can fly for very long periods of time. We're talking about hours. So that you can hover and go long distances and do this.

Now, this approach to data is called the data mule approach. In other words, for the Internet of Things to work, you need data to get to the Internet, and a data mule sort of collects data. It can do it, sort of collect the data, and then bring it back. Or it can hover, collect the data, and then transmit it back.

So those are some examples. Now, what I'm going to do is get a little bit into radio frequency in a little bit more depth. I could go on for days about this. I have spent years on it. But I want to give you a taste of what it takes to take an Internet of Things type technology from soup to nuts. Obviously, I can't give you the whole experience, but you'll get a flavor of it.

RFID Deep Dive (Sanjay Sarma): Video 2 Part 1

We have talked about RFID in the past, and now I'm going to take a slightly deeper dive and give you a sense of what it takes to make a technology go from an idea to scale implementation. As I've said before, there are quite literally billions of RFID tags out there, and it all started with some fundamental thinking about what is RFID? How do we use it? What's the architecture? The very same questions we've talked about during this course.

Looking back when we look at RFID, it's sort of interesting. We actually ended up with a three-tier architecture, which is very similar to the things we've been talking about so far.

And the three tiers are you need readers. Readers are the ultimate sensors. You need a middleware, sort of a local controller. And then you need the cloud, from which you share data. Because when we were doing RFID, the term "cloud" didn't exist, but we had to effectively sort of evolve that by using the Internet and servers on the Internet.

Now the one difference in RFID is the sensor is actually the reader. It's not the tag. The tag is a fiducial, because things don't like to talk. They can't talk. So you put a tag on the thing so that you can read it. So the tag is a fiducial to enable sensing.

And as I've said before, the three tiers are there's the cloud, there is edge intelligence or edge gateway, and then there's the sensor, which is the reader. And the sensee, of course, is the RFID tag. That's the thing you sense. So I just wanted to again clarify RFID tags are not sensors.

Over the course of this journey that took about 10 years to finish, we had to develop standards at every one of those layers. And one of those layers is between readers and tags. And that standard was the work of over 100 companies, users, technology companies, startups.

It was a pretty torturous journey in many respects, but also a fulfilling one. And I remain in touch with many of my colleagues from companies like Alien Technology, Impinge, and so on over the years who participated in this incredible journey.

So for example, if you just look at the layer between readers and tags, the standard there is the electronic bar code, EPC, which is a term that we made up, Gen 2 protocol. There were a couple of Gen 1 protocols which have been deprecated or they aren't used anymore. And now we have something called the Gen 2 protocol.

The official term for it, because this is also now an International Standards Organization, ISO, standard is ISO 18,006c. So you might sometimes hear EPC Gen 2, and sometimes you'll hear it referred to as 6c. It's all the same thing.

One layer up, you need protocols for software to talk to readers. And many readers have their own native protocols and APIs, but in fact, we ended up, the community actually,

ended up developing a standard for that, and it's called "Low-Level Reader Protocol," and it's a way to treat the reader like a black box, ask it questions, have information come back from it.

One level up you need the middleware to interact with the rest of the system. And for that, there was a protocol as well. My good friend and MIT alum by the name of Ken Traub pioneered this, and we called it ALE-- Application Level Events.

And then one level up above that, you have the cloud. And you want things to talk to each other across the cloud. For that, we developed something called EPCIS. So these are some examples of protocols.

Let's just focus in-- as a deep dive, we can't talk about it all-- about the reader tag protocol. Very interesting. RFID tags come in different flavors. You have tags with batteries on them. In fact, you could argue that your cell phone is an RFID tag at some level with a battery. Because it has an ID, it has a Sim card, but it also converts your voice into a signal. So it's an RFID tag at some level.

But the other end of the spectrum you have passive RFID tags. They don't have batteries. They scavenge their power, as I've said before, from the air. So we're talking about passive RFID, and that's what this protocol targeted, because we wanted to adopt RFID at scale.

The particular protocols that we developed operate in the 1860 to 1960 megahertz range. Why such a broad range? Because the same RFID tag has to work in the supply chain from China to the United States to Europe to India to South America. And over the course of many years, we opened up and lobbied governments to open up frequency bands to enable tags to work in all these frequencies.

In the United States, the FCC has something called an Industrial Scientific Medical Band, ISM Band. That band in the US is from 1902 to 1998 megahertz. And usually the way that these readers work is that the frequency hop on this band and they can't put out more than 4 watts, which is just 36 dBm Of EIRP power with a 6 dBi antenna.

And if you just Google EIRP on Wikipedia, you can read more about it. There's also something called ERP, but we'll leave that alone. And usually what we do is we hop over 50 channels and you can't linger for more than 40 seconds in a channel.

The reader to tag protocol is sort of the physical layer in RFID, and the readers have to communicate with tags. Now remember, tags are not very smart. They don't have RF computing power, because trying to make them cheap.

So we had to develop a protocol which, on the one hand, the reader could put out enough continuous wave signal to power the tag. So the reader has two purposes, to power the tag and then in that signal to modulate a signal to the tag.

Now if the reader, in the pod process-- imagine like a Morse code-- if it has to sort of go through long periods of silence, the tag might lose power. So it's a balance. And so what the community ended up doing in the Gen 2 protocol is using something called "Pulse Interval Encoding," PIE, in the reader to tag communication.

The way that works is that 0 is a short pulse and the 1 is a long pulse. And tari is just a reference time interval which we can talk about later. But the longer pulse means it's a 1. And the beauty of this is that if you have a short pulse and a long pulse sort of interfere, you can hear it.

The modulation from the reader to tag, the Gen 2 protocol accommodates several approaches. One is a dual side band amplitude-shift keying. There's a single side band amplitude-shift keying. There's a phase reversal amplitude-shift keying. So there's a whole bunch of different ways to communicate from the readers to the tags.

Tag to reader is a little bit more touchy. Remember, the tag is very dumb and the tags can't hear each other. The tag only communicates to the reader, and the reader hears a faint echo from the tag. Because a tag doesn't transmit, it's simply sort of like a mirror turning on and off. It reflects back a signal to the reader. When the reader stops talking, all the tags together stop talking back.

And actually one of the next things that readers do is figure out which tag is speaking, and then it's got to tell all the other tags to shut up so only one tag can talk. It's a very interesting sort of challenge in RFID. So the tag to reader a protocol is either FM0 or something called "Miller code." It's the same thing. FM0 and Miller are very similar and we use the FM0 protocol here.

And the way it works is that if you have a logic 0 in FM0, you get a positive transition from 0 to 1. If you have a logic 1, you get no transition. And regardless, you get a transition at the end so that you can show that there's a boundary between two 1's. And then let's say you have two 1's and it ends up on high. It has to transition back and then the 0 transitions up. And then it transitions back and so on. So that's the FM0 encoding from the tag to the reader.

RFID Deep Dive (Sanjay Sarma): Video 2 Part 2

Very interesting thing about RFID readers and RFID tags is because, in most communication systems-- let us say you're in a classroom, and say you're giving a lecture, and you have 100 students. When a student speaks, the other students can hear that student. But in RFID, the tags can't hear each other. They're really sort of, very limited in their capability.

So what you need to do is something called an anti-collision protocol so they don't collide with each other. So like, stop, stop, stop, you know, you can speak. It's almost like you're having a conference call. If everyone speaks at the same time, you can't hear anyone. But imagine if everyone could speak at the same time and they can't hear each other. That's sort of what RFID is, and so the host has to sort of figure out how to arbitrate and say, OK, you can speak, now you can speak.

For example, I could say, I'll start with everyone whose first name begins with an A. OK, only you speak. And then if two people speak up, I go, if your first name begins with an A, and the second letter on your first name is A, now you may speak. And if there's still a collision, I need to go to the third letter to arbitrate.

So that's basically what anti-collision is. So the RFID reader has to sort of sort them all out. Now, I'm going to now show you a sort of animation. It's not very detailed, but you get a sense of how anti-collision, or arbitration works.

So, one of the amazing things about the Gen 2 protocol is, imagine if you have thousands of tags. And let's say you had Gillette tags and Proctor & Gamble tags. And then you're Wal-Mart, and then you had tags on the next shelf that were from some, you know, Unilever. And they were on the wrong shelf. You just want to address the Gillette tags right now, because you want to see if the Tide detergent is out of stock. You've got to be able to say, listen, I just want to talk to the Tide detergent. I don't want to talk to everyone else.

So one of the elegant things about the Gen 2 protocol, is you can address a certain subset of the tags and say, listen I only care about you guys. And that's called a Select command. And once you do the Select command, you can take the tags that you care about, which are lit up in blue here, and you can sort of ignore the rest of them. And you can address only that subset of the population.

But remember that you sometimes may have a thousand RFID tags in your field. In fact, one of my students jokes, and he says, RFID tags are like cockroaches. I can't track them all down, and it's information noise. And I keep reading the same tag. So Select lets you say, I don't want to talk to you anymore. So that's the Select command RFID.

Now, once you do that, you can take those tags and you can say, I want to inventory those tags. And the way that works is the reader now goes into this mode called Inventory, and you want to Inventory. And the reader sends out a query to these tags.

And it says to them, look, I want to talk to you. And for those of you who know this, it's called the Slaughtered Aloha program. So if you know Mac protocols, it Slaughtered Aloha.

So it says to all the tags, I want you to pick a random number. But one of the things the reader does is it has a sense, and there are a lot of algorithms for doing this depending on the use case, of how big the population is. Is it just four tags, or is it 1,000 tags, right? So what the reader does is it gives the tags-- it sends them a parameter called Q. And this is called a Q algorithm. It's a wonderful invention of a few people in the EPC community. Kristie [INAUDIBLE], Steve Smith, a whole bunch of these folks-- really brilliant folks worked on this.

So you send out this Q, the number. And the tags then pick a random number between two to the power of Q, minus 1, and 0. So, for example, 1 and 3, 0 and 3, 0 and 15, something like that. If you think there are 5 tags in the field, you can say between 0 and 15. And picking the right Q, there's a lot of algorithms for it. And, as I said, you can do it based on the population you expect.

So, now let's say these tags have picked their slots. So this tag picked 23, this one, 14, this one, 1, this one, 10. But this one picked 0. Awesome, now you have one at zero. And so you go, all right, Mr. Zero, you'll go. Respond. And if none pick zero think cycle, every one of them decrements by one until one, or two, or many of them have zero. Now, in this case, only one hit zero. I'll come back to when two hit zero in a second. I'll tell you what happens. That's called a collision.

Now, when this tag hits zero it responds by something called backscatter. The tag, as I said, now transmits. It's sort of a mirror that turns on and off. So it backscatters the reader's own signal back to it. But also sort of using the reader's signal to power itself. It's very fascinating.

And it sends a random number, which is 16 bits. And that becomes a temporary identifier for that tag. And then the reader sends back a signal to the tag. It acknowledges it. And the tag goes, is that a valid signal?

Now here's the deal. Let us say that two tags responded, and the reader only heard one of them. The second tag is going to say, a-ha, that wasn't a valid response. So it goes back into arbitration state. It goes back to the state. But the one that was identified, because it was louder, says, I'm ready. I'm acknowledged. And then it can go into the state where it sends its other information.

It sends numbers, and it sends something called a CRC, which is sort of like a-- some CRCs, is likely a redundancy check number. Which, when the reader gets it, the reader knows, OK, I really got a tag. Because not only did I get its numbers, but I got a confirmation on the end. Which, if the numbers were wrong, I could've picked up that the confirmation would have been wrong.

And now the reader can say, I'm going to go into access mode. And now you can access the tag. It sends access commands, the tag is open. You can get other information, you can write information, et cetera.

So all this is to do three very simple steps. And this is just the beginning. There's actually a lot more. You can write to tags in bulk. You can kill tags. There's anti-counterfeiting. It's a very complex protocol, and I've given you the most basic introduction to how RFID readers and RFID tags work. And what you may have seen here is how tightly things are coupled to make it work. It's almost magical, how these technologies work.

And the subtleties here are, there's a Select, there's an Inventory, and there's an Access. Tags can be killed afterwards. Now, later on, later versions of RFID, we've added some other functionalities. Even in the earlier version.

Imagine if two readers were hammering at the same tag. Would the tag not get confused? Imagine two teachers in a classroom with the same kid. The kid can get confused, right? But RFID tags in the Gen 2 protocol have multiple sessions so they can talk to one reader, and also talk to the other reader. Very fascinating.

And then there are also some other things they can do. So, they can work in Dense Reader mode. So, imagine if you had hundreds of readers along an aisle, reading. RF signals are not like light. They don't go in straight lines. They bounce, they sort of go in different directions. So the same tag can find itself being read by different readers. And this protocol will survive that.

And all sorts of other subtleties. So for example, imagine if you had a warehouse, your competitor's warehouse. Now, readers have to really blast out their information, all right? Because they have to talk to these tags which are both getting power and signal from the reader. Imagine if you parked a truck outside with a listening device, and you could figure out all the inventory inside the warehouse, your competitor's warehouse. So you can say, hey, they've got the new order of iPhone 6, and we need to move, right? It's comparative information.

So one of the beautiful things the community did-- and this is really a truly community effort-- is they put as much secret information as possible. Not secret, but, you know, information you don't want on the forward channel. They put it on the back channel. And what that does is it makes it difficult for someone listening from the outside to hear, because the response of the tags is much weaker.

So all this is a subtle, beautiful protocol which addresses a really staggering range of issues. And in the new Gen 2 v2 protocol, tags can be used to detect counterfeits. It's from a strong encryption. You can control access, so only some people can access certain information. Certainly you don't want anyone to be able to kill a tag, otherwise a bad guy could go into a store, kill a bunch of tags and steal those products.

You can also predict privacy in some ways. You can make sure that my tag can't be read by someone else-- maybe when I leave the store.

One of the things with RFID in the early days was valid concerns about privacy. Now, the good news is the early implementations of RFID were mostly in the back end of the supply chain. And now, with some of these additional benefits, should someone want to detect-- read what you're buying in your groceries, they can't. It's hard to do it anyway, but now you can protect against it.

RFID Deep Dive (Sanjay Sarma): Video 3

We've learned a lot of lessons. This is just one journey. Every wireless protocol, every protocol, every engineering system-- whether it's the internet, whether it's telephony-- is a long journey that involves hundreds of people and thousands of little issues and addressing each one of them as best as you can. It's sort of a harrowing thing, frankly. But one of the lessons learned is building a working, secure system is difficult. It is very difficult. It takes blood and sweat and tears.

It's a lot of IP involved and legal stuff and technical stuff and breakthroughs on the chip side, breakthroughs on regulatory sides, et cetera. Starting from scratch is very difficult. And IP, I mentioned earlier, is becoming today an even bigger minefield because of the litigation around IP. You have companies that accumulate IP and sue other companies, so-called non-practicing entities, NPEs. And so IP is more and more a minefield. So it's hard to start from scratch.

And the future, from the RFID story, the lessons for the future for IoT are, must reuse existing standards as much as possible. It is very difficult to start from scratch. It's exponentially more difficult now than it was when we started with RFID in 1998. The architecture is so profoundly important. A mistake in the architecture will cause pain for decades. Must think about security right in the beginning. And end to end security, very powerful.

And you must think about maintainability. There's a sort of a honeymoon period from your first implementation to your first breakdown. And it's OK when you're implementing, but when someone needs to go in and fix it, that's when maintainability becomes an issue. And you've got to think about it. Security and maintainability, I think, are so important. I'll talk about it more in the conclusions again, just to emphasize the point.

More lessons. You know, when we started the RFID story, a lot of folks thought, oh, it's all going to big data. We'll get lots of data. Not really. It's actually narrow datas, lots of chunks of small data. It's a different thing. And they tax existing systems in ways that they didn't expect. Thousands of little, sort of writes into a database, and database technologies had to keep up with it.

And the data is very pointed. It's that reader, and that doctor read that pallet. And the data has errors in different ways. So for example, maybe, maybe, just maybe, the pallet was sitting, but the tag fell off. So you need to repair the data in different ways. Don't assumed that IoT is about big data. It will happen, but in the early days, the data is just different. And the modalities are different.

We also thought, oh, it's all going to be about networking standards. Well, yeah, to some extent. But power over ethernet was something that a lot of RFID readers. Used. Now a lot of readers have gone to handhelds, and they're battery powered and Wi-Fi. It's more

about the architecture. You can abstract away some of the details of the network. Wireless protocols, the next thing I'm going to say, are highly specialized.

I'm often asked, do you think that the BLE will go away, and ZigBee will go away, and there will be one protocol, one ring to rule them all? I don't think so. The reason is, wireless protocols are real small miracles. And they all work very uniquely in one situation or another. You just saw the RFID wireless protocol. Because the tag has no battery it's very unique. I couldn't have used Wi-Fi on RFID tags. That wouldn't have worked. We couldn't have used BLE, because they don't have batteries.

So BLE serves a very interesting niche. ZigBee serves a different niche. Z-Wave serves a different niche. And they all have different purposes. My phone, my iPhone or my Android phone has BLE. It doesn't have ZigBee. So be ready for a bunch of different wireless protocols, of a gene pool of wireless protocol to survive, again, use abstraction. So that it almost doesn't matter.

Another lesson is, one of the things that folks did was they said, oh, RFID is just a replacement for a barcode. Wrong. RFID is not a replacement for a barcode. When you pick up an object-- I'm going to pick up my cup of water here, I'm going to imagine there were barcode-- I would have to take a reader and scan it. But RFID is not that polite. RFID reader is going to read that cup while it's sitting on the table. And it might read a cup in the other room. They're very, sort of, obstreperous.

And so you can't just plug them in like a barcode reader. It's a very different thing. It's silent intelligence, to use a term that was marketed a few years ago. It is just sensing stuff all the time. It is the world talking back to you whether you like it or not. It's not, hey, talk to me now. It's like kids, they're just talking all the time. And you've got to make sense out of that. And so it is not business as usual. It's a whole new business, a whole new sort of way of dealing with the world.

And that was a huge mistake that a lot of companies made. And it was frankly promoted by companies that had barcode reader software, right? They were like, ah, just the same thing. It's not. The same thing applies to internet of things. It's a whole new use case. It's a whole new workflow. Take advantage of the new workflow. Don't assume it's the same old thing and plug it into an existing system, which will give you a narrow view the world where you lose a lot of the great things that this technology enables.

RFID is still, 15 years later, in its early stages. Given you a couple of examples, including the introduction of how RFID can be used for sensors. My view is because it is cheap and throw away, people will have an array of RFID tag-based sensors-- temperature sensors, which I've discussed. I just showed you a blood sensor. You know, pressure sensors. There's going to be an array of things, not just ID, but beyond ID. So that is a very important topic of research.

Now we're doing a lot of it. Others are doing work all over the world, actually. And this is an exciting area, because imagine throw away sensors. It's a very powerful concept.

Another topic of research is the readers today in RFID still consume power quite a bit. But imagine very ubiquitous readers, but at very low power.

Already there's some very interesting companies that have handheld readers. You have readers that can plug into your cell phone. I came across a company recently called Crockett. Just plugs into your cellphone, and turns your cellphone into the computing interface to a reader. And then your cellphone provides a connection to the cloud. So the cloud architecture works there as well, by the way. It's sort of interesting. That's where I'm a big believer in the cloud in this whole space.

But reader proliferation, reader modalities, is a huge area of research and development. I think we'll see a lot of innovation there. Another very interesting thing, RFID tags are good for sensing the presence of an ID. But what if you could use them also to get range and bearing, that is, location? Already some readers are doing that. In fact, quite a lot of them do it now. So not only can you say, hey, I see that ID, but I see it there, and it's about 3 meters away. Because that enables, again, some very interesting modalities. And that's an interesting area of research.

And sort of related to that is what I call RFID calculus. So let me explain what that means. Let's say I have a reader in this room, and I can read tags in this room. When the door opens, a reader outside can read the tags. I go, oh OK, the door opens, maybe it read it inside this room. But maybe I see a tag here and then I stop seeing it, and then a reader there reads the tag. Well now I just inferred motion. It went from this room to that room.

Let's say one reader facing this way sees it, but another reader facing this way does not. Now I know it's on this side. So I think we will, in addition to range and bearing, double up the whole calculus around RFID to infer location, info status, and so on. And now if you add temperature, et cetera, sensing, it's a whole new field of mobile sensing that you're evolving.

And then finally, I think we will see an incredible convergence of technologies-- RFID readers that are built into something like Google glasses. So I see the product, and maybe my display shows me-- let's say I'm maintaining some piece of equipment-- I'll read it, my augmented reality shows me that piece of equipment, it's drawings, but also its maintenance history, right? Or watch out for that part. It's wearing out, right?

It could also be, for example, vision, RFID barcodes, it could be image recognition-- we're actually doing a lot of research on this. And so we're entering a world of multimodal perception. And RFID becomes an anchor tenant in that multimodal world, because it provides confirmation of ID. So these are some of the amazing things that are going to happen. I'm going to stop here, but you're going to see me in the conclusion section, sort of wrapping it all up.

I hope that you found this useful. It's like a deeper dive into one particular vertical technology. It's not very deep, really. But, and if you're interested, you can read about it. There's a lot written about it. But I hope you've also seen that some of the things we've

talked about in this course, the lessons are hard learned. They're sort of not something we just made up. And we hope that as you begin your journey into the Internet of Things, you will heed them, and we hope you'll be successful in them. Thank you, very much.

Week - 3 Module Three: Technologies

Network Connectivity for IoT (Hari Balakrishnan): Video 1

Hello, and welcome to this module on Network Connectivity for the Internet of Things. My name is Hari Balakrishnan, and I'm a professor of computer science at MIT, and I lead the Networks and Mobile Systems Research Group at CSAIL. I've worked in the general area of network computer systems, and in particular, wireless mobile computing and sensor computing for over 15 years. And this is one of my major research areas, so I'm excited to tell you about networking for the Internet of Things today.

The flurry of activity is because of technology push on the one hand, and demand from applications on the other. And when you think about the technology drivers for the Internet of Things, there are three things to keep in mind. First and foremost, number one is embedded computing. Second, because of advances in MEMS technology and miniaturization, we have a variety of sensors and actuators that are now possible to put into a variety of small devices and put all around the environment. And third, wireless communication networking. You should think of wireless communication and wireless networks as the glue that hold Internet of Things systems together. And that's what I'm going to talk about today.

If you look at network technologies for our servers, and laptops, and desktops, and smartphones, over the past 40 years or so, we have developed a good understanding of how to design them, and how to engineer them, and how to manufacture them. So these are fairly standard technologies. Of course, there's a lot of innovation and evolution in networking for normal computers and mobile devices, but there's a pretty good way in which we can predict that evolution, how they're going to evolve into the future.

When it comes to networks of devices and sensors and the Internet of Things, that's not true. This field is much newer. There's a lot more excitement happening. There's a lot more confusion in the field.

And what I've shown this slide here is a wide range of different technologies possible. You can have Bluetooth and Bluetooth Smart, ZigBee, and 6LoWPAN, IEEE802.15.4, 3GPP, Lte, various companies in the mix. And now you can see lots of different emerging standards out there. In fact, there's four standard bodies that are listed at the bottom of this slide. In fact, I don't have enough place on the slide to go through all of the different commercial products, research efforts, and standards proposals that are out there. And the ones I'm showing here are not really one that I particularly am endorsing-- although there are some excellent technologies in here-- but this is just to show you the connectivity soup that's out there.

So what is our goal for today? Our goal for today is to cut through a lot of this confusion. There are a large number of different approaches, a large number of different standards, and frankly, a lot of confusion. One of the main messages for today is that one size does

not fit all when it comes to IoT network connectivity. The best choice of network depends on the application.

And so my focus for today is to explain to you how to think about this field. What are the key organizing principles? And what are the main ideas that you will benefit from whether you are designer, or engineer or IoT systems, whether you're an architect of such a system, or whether you're managing the engineering or deployment of such a system. So that's what we're going to be talking about today.

So I want to spend a few minutes telling you a little bit about my experience with the Internet of Things. Now I joined MIT in the late '90s, and in 1999, soon after I joined MIT, I started a search project called Cricket. It's an indoor location system. Think of it as an indoor positioning, indoor GPS system. And the technology works using a combination of radio frequency and ultrasonic transmissions.

These little devices called beacons you would put on the walls and ceilings of you're building, and they would advertise beacon messages on both RF-- radio frequency-- and ultrasound. And receivers with similar hardware would measure the time difference of arrival between the time it takes for the radio signal to propagate and the time it takes for the sound signal to propagate. Remember that sound travels a lot slower than light. And using that time difference of arrival between the signals, they would estimate the distance from these different beacons in infrastructure.

We also engineered ways by which these beacons could self-configure it into a coordinate system, and we could use that to build, very accurately-- centimeter accurate-- indoor location system. This technology was available under open source, and was also was commercialized. And even today, there are niche applications in places like hospitals using the Cricket system.

From 2005 until about 2011, with my colleague Sam Madden, I lead a project called Cartel, which was one of the first mobile sensing systems. The idea here was to think of your cars and automobiles as sensor networks, put in embedded computing and sensing capabilities in your automobiles, and drive around collecting interesting data. And we use this to build what today is known as the connected car, and focused on a variety of mobile sensing efforts. This included new ways to connect vehicles with Wi-Fi, and use vehicular Wi-Fi to connect to the internet, new ways to think about mapping and map matching technologies. We built traffic of our routing systems in the city of Boston and Cambridge, and also built a pothole patrol by which you could use these automobiles to detect potholes automatically and create maps of where the hazardous road conditions were in the Boston area.

Since about 2010, again with my colleague Sam Madden, I've worked on a system called Drivewell, which is actually commercially available now. Drivewell is an effort in smartphone sensing focused on safe driving. So we built smartphone systems, as well as an embedded IoT device, which has some sensors on it that you attach to the windshield of the car. And the smartphone, together with this device attached to the windshield,

provide information about measuring how safely you're driving as a driver and provide incentives for you to become a better driver. And one of things is device does is also detect accidents and detect road crashes in real time. So by doing collision detection, we can have roadside assistance come to assist a driver who might be in a road crash using IoT technologies.

So the focus for today, as I said before is, how to make sense of this connectivity soup. And I'm going to draw on the experience in our research efforts at MIT, its commercial efforts, and the wealth of literature and research activities are happening in the field today. And I'll talk about how industry standards are evolving in the space as well.

Network Connectivity for IoT (Hari Balakrishnan): Video 2

So how do we cut through the complexity of all of the IOT networking options out there? So we're going to start by simplifying the architecture.

So if you take a look at this picture, what you see at the bottom are these things, these devices and sensors and actuators. These things have data that we'd like to deliver to servers, and to applications, so we can process the data and understand what they're saying about the environment. Of course, we have the internet. Which is how we're going to get the data to the places where we're going to do various competitions. Now the main thing to understand in the simplified architecture is that in order to get the data from these devices-- these sensors-- to the servers in the cloud, we're going to need gateways. These gateways could be static devices that you put out in your homes, your offices. Or out on the streets, or on the road, or they might be mobile. They might be your smartphone or tablets. Things that you carry around, people carry around, can also function as gateways.

The job of the gateways is to mediate and translate between these things and the internet. So what we're going to be studying are ways in which we can connect these things together to the gateways. The gateways will then do the translations and mediations necessary, so that these things can become, essentially, first class citizens in the internet.

Now why do we have all this complexity? We showed you a very simple to understand architecture. So turns out this is, in fact, a very rich design space. The question of how should gateways and things communicate has many, many answers. But the best answer depends on the application. There are many answers, and many approaches, and that's what we're going to talk about. Before we dive into understanding networking for the IOT, one question that you should have in your mind is, what are we just use the wireless technologies that we have for the internet-- which we have on our cellphones-- to build IOT systems? Can't we just use cylinder networks and Wi-Fi technologies? Why do we need something new?

And the answer, to paraphrase President Obama from 2008, is yes we can. We actually can build IOT systems like that, and there are many examples of IOT systems built this way. But it turns out that there are many, many places where we just can't do that. And I want to explain to you first how we can build IOT systems with cellular and Wi-Fi technologies, and then explain to you why, in many cases, that just is not a tenable option.

So one of the common wireless internet options out there-- whether they're cellular technologies like LTE or 4G or 3G or even 2G, and there's Wi-Fi. Both of these are widely available, and support high bandwidth, which means you can build high data rate applications with them. So what are the drawbacks? Each of these technologies has two significant drawbacks. First, both technologies have very high power consumption. What that means is, you cannot apply them to battery operated IOT scenarios. And many scenarios are battery operated, where we might have a device that we put out in the field,

and you want it to run for days, months, or in many cases even for multiple years, on a smaller battery.

Now cellular has another drawback. The major drawback of cellular technologies is that often it has high cost. You might have to spend maybe \$1 or \$2 a month for each device that you've connected your system. This is not a useful option if the amount of usage, the number of bytes being sent by your thing per month, is not very high. So you're spending a lot of money per month, but the amount of data that you're getting out of it is not that high. We might not have a lot of data to pull out of it. So cellular is not a cost-effective option in many cases.

Now, Wi-Fi doesn't have a cost problem, Wi-Fi has a problem of extensibility-- that is to say, range. Wi-Fi is OK in most buildings and campuses, but not when you want to cover a longer range, like kilometers of range. But that said, Wi-Fi is a pretty good option for IOT when you have powered things inside buildings. So, for example, speakers, or if you want to connect your washers and dryers in your home, or refrigerators. To build an IOT system, Wi-Fi is a pretty good option. And cellular is a pretty good option when you have a high value, powered thing. In fact, it's the common option today for so-called connected cars. Usually there's not a problem with power in your car, if you're connecting to the onboard sensors in the car. And that's a good way to build an IOT system using cellular technologies. And later on in this module, I'll talk about what are some interesting problems that arise when you want to build a very high rate cellular IOT application.

So I want to spend a little bit of time telling you about cellular power consumption. So what you see on the left here, is a picture of a device called a power monitor, which is what engineers and researchers use to measure the power consumption of devices. So you attach the power monitor to the leads of a battery, opening up your phone or other device. And then you run applications on the device. And what you see is a plot, as I've shown on the right here, where you see time going on the x-axis. And on the y-axis, what you see here plotted is the power consumption. And what the application here's doing is-- it's quiet for some time and then what it's doing is-- communicating over a cellular network. And then it's going back and is being quiet again.

And what you see is something very interesting. You see different levels in this power diagram. You see a portion here, where it's relatively low power and is flat, and comes back to being flat here. And that corresponds to the radio being idle. There's very little activity happening. In fact, all of the power drawn here, most of it, is other things that are happening on your phone. Now then, the second thing you see is a spike that happens when communication starts happening. And in fact, if you look at the entire period of communication when data was being sent back and forth, that's shown here highlighted in blue. That's the portion corresponding to data transmission.

Now what is the rest of the stuff? The rest of the stuff you could think of as just waste. And what the phone is doing here is going through different kinds of idle periods, but all of that is consuming power. And it's divided into two parts. There's a certain amount of time where the phone remains on in the highest active state, thinking that there might be

further communication happening, so you don't want to shut it down. And then it goes into a different state, called the High-power idle state. It's actually idle, it's not shut down, but it doesn't consume a small amount of power. It consumes a moderate amount of power. And it's because of this that if you run an IOT application in your cell phone, and all you're doing is sending some data every 30 seconds, you still end up consuming a huge amount of power. And what ends up happening is your battery may last maybe four or five hours, which is just not good enough for many IOT applications. This fundamental problem of power consumption is why cellular and Wi-Fi technologies are not applicable to a wide range of IOT scenarios.

Network Connectivity for IoT (Hari Balakrishnan): Video 3

We just talked about power consumption in the cellular context, and what I'm going to do now is to tell you about the IoT network design space. So this is one of the key lessons of today's module here, which is understanding how to deal with the complexity of the different IoT options.

So what we're going to do is simplify our description down into three axes so that you can think about any given IoT network option along these three dimensions. The first dimension is battery life. In fact, this is the reason why we cannot always use cellular or Wi-Fi technologies.

So the battery life depends on the technology. Now, on the one hand, on the left extreme here, you might have a powered device, in which case battery life is really not a concern. So this is what would happen, for example, in settings where you can attach your devices or sensors or IoT things to a power source. So you don't care about battery life, then.

But on the right extreme here, you might have an IoT system which is deployed, for example, in an industrial setting or in a field setting-- for example, environmental monitoring in forests-- where you might want to run for 20 years. In fact, these are realistic time frames, that people are trying to build systems to achieve 20 years of battery lifetime. And you have lots of things in between.

So the first axis is battery life. The second axis is the device's data rate or duty cycle. So what I mean by this-- think of it as the number of bytes per day that you would like to communicate between your devices and the servers on the internet.

So you could go as little as bytes per day, or even a few bytes every few days-- which is at the lowest extreme, is bytes per day-- all the way up to gigabytes per day. So you might have applications, for example, capturing images from cameras, still images or videos, where you might have gigabytes a day. So the second axis is the duty cycle of the data rate of your device.

And the third axis is the range of communication between the device and the gateway because remember, once you get to the gateway, it can run software to mediate and translate to the rest of the internet in your servers. So it's all about how far your devices are from the gateway.

And here, the ranges extend from inches-- such as might be the case in payment systems, where you take your smartphone, use Near Field Communication to pay for something you just bought-- through Body and Room-Area Networks, where you want to interact with devices that are within the same room-- to buildings and factory- or campus-level settings, all the way to miles of connectivity. So again, the third axis is the range between the device and the gateway.

Now, if you look at these different axes, on any of these dimensions, we're talking about many orders of magnitude. So the message here is that no single IoT system solves the problem across the entire range on these three dimensions.

But what's really nice about this way of thinking about it is any given technology or standard, you could put into this axis here and think about your application, think about your system on these three dimensions and say, well, number one, you could look at a technology and say, where does it fit in this dimensional space? Or if you were designing an application or thinking about a deployment, you could say, well, what are my requirements on these three axes? And based on that, I can pick the best networking technology.

So where do Wi-Fi and LTE cellular technologies fit into these axes? Well, Wi-Fi has megabits per second of peak data rate, so you can run gigabytes per day. It has a battery life of a few hours if you were to run such a duty cycle. And in terms of range, you could apply it to anywhere between a Room-Area Network to typically a factory or a campus.

And with LTE, you get a little bit less than Wi-Fi-- maybe three or four hours. Again, you get gigabytes per day, if you're willing to pay for it, and you get miles of connectivity.

So now, why are there so many IoT networks? Well, a glib answer is because engineers love inventing technologies. So we have lots of different options. But really, it's because you can pick many interesting regions from this three-axis design space.

In fact, note that the axes aren't independent. For example, you might pick a given network technology. And if you were running it at a duty cycle of 100 kilobytes a day, you might have the batteries last you two years. But if you took the same technology and run it at gigabytes per day, your battery might only last a couple of days. So these axes are not independent, which makes the design space even more complicated.

And the other thing is technology evolves really, really fast. In fact, this is true for networking technologies in general, but it's especially true of these small embedded devices because advances are happening pretty rapidly.

And one must not underestimate the effect of bundling technologies into popular devices. These speed up adoption and change the economics. So things that didn't look so tenable just three or four years ago all of a sudden become widely popular and become the way to do things.

So for example, this is what happened with Wi-Fi technologies about 15 years ago when they started getting embedded into laptops. 15 years ago, you wouldn't be able to get a laptop with Wi-Fi. You started off attaching wireless LAN technology as external cards, and they started becoming a little more popular and more popular. And soon, the laptop manufacturers started putting it onto the boards, and then it just exploded.

The same thing happened with Bluetooth classic. This is the kind of Bluetooth you use to connect your headset to your phone. The technology is actually quite old. It was conceived of in the mid-1990s, but it took about 8 or 10 years to understand what the key killer application for that technology was.

And it turned out to be a replacement to the cable that connected your headset to your phone. And when that killer application was identified and all of the cellphones started having Bluetooth on them for that application, the technology took off.

And most recently, this has happened with Bluetooth Low Energy, or BLE, which we'll talk in detail about because today it is one of the most popular ways to do roaming area Internet of Things technologies. Initially, the iPhone and then Android smartphones have embedded Bluetooth Smart technology or Bluetooth Low Energy technology in them. And now all of a sudden, because you have it in the phones, you now have a natural gateway.

These mobile phones can act as gateways to these devices that have Bluetooth Low Energy. And all of a sudden, we've solved the problem of getting the data from these embedded devices because all of the phones have Bluetooth Low Energy. And now you can build applications which have months to years of battery lifetime with relatively low duty cycles because the technology has got embedded in phones.

So the reason why there are so many IoT networks is because you actually have a very rich design space, technology evolves rapidly, and the game changes when certain technologies get embedded in popular devices because now that opens up a lot of options.

Network Connectivity for IoT (Hari Balakrishnan): Video 4

Now we're going to be talking about a few different case studies. The first one we'll start with is Bluetooth Low Energy, which is an example of a Body/Room-area network. So again, in our three axes here's what it looks like.

So you have the device's data rate. The peak data rate of Bluetooth Low Energy is about 100s of kilobits per second. We translate that into number of bytes per [? data ?] [? moment. ?]

In terms of range, it's body to room. So the typical range might be about 10 meters. Now under very good conditions, you might get 50 meters of range with a high transmit power.

Remember that a longer range might not always be a good thing. You might want to build an IoT system where the range of communication is within a room or within an automobile. And you don't really want to extend further outside.

And on the dimension of power consumption, you can get months to, in fact, multiple years of battery life if you were restricting yourself to just a few hundred kilobytes per day of communication. And, of course, if you were to change the amount of communication to be gigabytes per day, you could do that. That would reduce the battery life.

A little bit about the history of BLE. It started as Wibree by engineers at Nokia, and has today become the dominant technology. And the reason it's become dominant is because of smartphones.

As I mentioned before, iPhone and now many smartphones embed this technology inside them. And what that means now, is you have a very natural gateway. Smartphones acting as gateways.

So now you can build wearables, like things to monitor your fitness or to track your activities. And also put them in vehicles, like bicycles and automobiles. So how does BLE work?

So there are two parts here that you have to keep in mind. The first part-- these things that have BLE technology on them, devices and sensors, advertise. These advertisements are also called beacons.

And they advertise that they have data to send. And the second part are that gateways and other devices can connect with these things that are advertising and exchange data. So there are two distinct phases-- advertisement and connection.

Now the technology for how these devices communicate, you should think of the devices as in one of two categories. There are central nodes and peripheral nodes. A central node, here shown in this picture, [? economical ?] example is a gateway.

A phone, in this case, is a central node. Peripherals are these devices that have data that they want to send. These are the things that we'd like to connect to the rest of the internet.

So the peripherals are devices with data, centrals are the gateways. The peripherals are the ones that are advertising or beacons. And the central listens to these advertisements and then initiates a connection with the peripheral in order for the communication to occur.

Now advertisements in BLE are typically periodic. So an example period might be every 100 milliseconds the thing, or the device, advertises its existence. So what I've shown on the left here is an advertisement from my Apple Watch, which is a BLE device.

So normally this advertises every 100 milliseconds. But less frequent advertisements are fine, and often desired if you would like to increase the battery life. So for example, if you were to change the advertising frequency from 100 milliseconds to 10 seconds, then what you would end up with is a longer battery life.

But then you may end up taking much longer for central nodes to discover the existence of the device and communicate with it. So again, the right periodicity of advertisement depends on the application. The other thing to note here is that triggered advertisements are often a good idea.

So for example, the picture on the right shows this tag that you attach to the windshield. This device only advertises when the car is moving. The rest of the time, an accelerometer on the device tells you whether the car is moving or not.

If it's not moving, you don't have to waste your energy advertising. And if you apply tricks like this and techniques like this, you can actually build systems that have multiple years of battery lifetime, even with BLE technology. Once a connection has been established between a central node and the peripheral node, the central node can discover a variety of things about the device.

And the way in which BLE organizes the information for what data is available is hierarchical. So each device has a set of services. Each service has a set of characteristics.

And there is a protocol that the central can engage in with the device that allows it to discover these services and the characteristics of the services. And there are various kinds of characteristics. Some of them are only readable.

So for example, you might have a sensor which is a heart rate monitor. You can only read the heart rate out of it, and you're not actually allowed to write to the characteristic. Because you don't want applications changing the heart rate that's been reported.

Certain other characteristics are both readable and writable. And certain other characteristics are notification characteristics. Where, if the data changes, then the central node gets notified that the data has changed.

And usually all these devices support some form of over-the-air programming or over-the-air upgrades. So if you want to change the software on the device, you can have a service and a characteristic that supports programmability. So in order to upgrade the software, you don't have to bring the device back to the factory or take it to your computer and connect to it.

You could do it over the wireless network. Now how does the protocol work once a central and peripheral node have connected? And this is a class of protocols called Media Access Protocols or MAC Protocols.

In BLE, the central node orchestrates all data communication. And the key idea is to time-schedule all communication to reduce energy consumption. So the central node and the peripheral node agree on a time period with which they will each wake up and send information.

In particular, the peripheral will be asleep the rest of the time and only wake up on an agreed upon schedule. And the secret to maximizing or increasing the battery lifetime is to sleep most of the time. And that's what we do in BLE.

The way it works is on a connection, the central and peripheral node exchange parameters. They exchange a frequency hopping sequence, which is the set of frequencies that they will each hop in a coordinated way to communicate. And they exchange something called the connection interval, which is how often they communicate.

So for example, T here, which is the periodicity of data exchange, might be 20 milliseconds or 50 milliseconds or 100 milliseconds. Which is how often they will wake up, communicate, and then go back to sleep. So every T milliseconds in BLE, the central and peripheral can exchange anywhere between one and four packets, alternating turns.

So that's how the coordination works. There's no overhead once you establish the connection and you exchange the parameters. And between these exchanges of data, the rest of the time the peripheral goes to sleep.

I want to explain to you how the BLE system allows us to build IoT devices that have multiple months, in fact, multiple years of battery lifetime. So I'm going to do this through a simple example to work out an energy calculation. This calculation will give you a flavor for how to go about doing similar things in other contexts.

So consider here an IoT system with coin-cell battery-powered nodes. A typical coin-cell battery, a small coin-cell, might have 250 milliamp-hours of capacity and might run at 3 volts. So milliamp hours is what we think of as capacity.

You could think of it as an estimate of the energy. Now recall that the power is the product of the voltage and the current. And the energy is the product of the power and the time.

So if you combine those facts together, this battery has 0.75 amp-hour volts, which is the amount of energy that it has inside. And if you were to convert that into SI units, one amp-hour volt, exactly 3,600 Joules. So what this battery gives us is about 2.7 kiloJoules of energy.

Now if you were to build a BLE device, here's an example of the current draw. The amount of current that might be drawn in a device like this. Today's BLE devices typically have current draw in the one to 10 microAmp range.

Let's take, for example, one microAmp. We have hardware today that when they're receiving, the current draw is 3.3 millamps, and when they're transmitting the current draw is four millamps.

So given these facts, suppose we have a device that transmits every second. Every second it tells us that it has some data. How long does the battery last?

So that's what we're going to try to calculate. So what this slide shows is, on the left, I've just copied over the statement of the problem from the previous slide. So there's nothing new there.

And on the right, I've shown pictorially what is going on. The x-axis on this picture is Time. And the y-axis on this picture is the Current Draw in millamps.

And if you were to draw this picture using the facts shown on the left, in the idle state here when nothing is happening, there is one micro amp of current draw. That's what happens in standby mode. And then there's a ramp-up that happens, where for one millisecond there's four millamps of current that's drawn when there's transmission.

And then there's a ramp-down that's happening. Now I have to say why we picked one millisecond here. We picked one millisecond here because of an assumption that we made.

It's a reasonable assumption, that when we communicate, we transmit 125 bytes of data. Just a small application so we send a small amount of data. So we pick 125 bytes, which is 1,000 bits.

And if you round that at one megabit per second, which is the peak rate of Bluetooth Low Energy, that's about one millisecond. So if you look at this picture here, there's one microAmps drawn in the standby state. And then we have a ramp-up, and then we have four millamps, and then we have a ramp-down.

So let's, work all of this calculation out. The way we're going to do this is divide time up into one second boundaries, and look at the total energy drawn in a one second period and divide by a one second interval. And that allows us to then calculate the average current drawn in every one second interval.

And using that average current drawn every one second interval, and knowing the milliamp hours of our battery, we can calculate the battery lifetime. Now, the ramp-up and ramp-down take some time, so we're going to have to make some assumptions. We can do a measurement with a power monitor.

And when you do that on a typical device, you may find that it takes one milliamps for five milliseconds. So if you take that together, and you work the numbers out over one second interval, if you calculate the average current drawn, it's four microAmps during the transmit. That's the four milliamps drawn for one millisecond.

You add to that five microAmps during the ramp-up and the ramp-down period. And you add to that a one microAmp current draw during the standby. And that's the total amount of current drawn, on average, during a one second interval.

Now, therefore, the battery lifetime you cannot calculate. Because you take 250 milliamp hours, which is your battery capacity. You divide by 10 microAmps.

And when you calculate that, that works out to 25,000 hours, which is about two years and nine months. So if you're on an IoT system on BLE with these parameters, which are all very realistic parameters, what you get is roughly three years of battery lifetime. The reason this all worked out is what I said before.

That that's because the device is sleeping most of the time. And that's really the key to why BLE works. Time division scheduling allows us to coordinate on a connection.

The device sleeps most of the time. And it's also sleeping and consuming a very small amount of power between these advertisement periods.

Network Connectivity for IoT (Hari Balakrishnan): Video 5

We talked about battery lifetime, and went through the case study of Bluetooth Low Energy, which is a great example of a low power IoT network. Now we're going to talk about how to extend the communication range. And again, we're going to go back to these three dimensions of the IoT network design space.

Battery life, the device's duty cycle or data rate, and the device to gateway range. We went through the inches and the body room networks. And now we're going to look at how to build an IoT system in the building scale or the factory or campus scale.

The technology we're going to use to do this, is now known by the term mesh networks. And actually is an area of research that was pioneered in the '70s and '80s funded by DARPA in the program called the Packet Radio Network Program. And there were many papers written on how to build communications systems where devices could communicate with each other, not directly to a gateway or a router to get to the rest of the internet, but have multiple hops from device to device.

Go through multiple wireless hops before you get to a gateway and get to the internet. So the natural way in which you might extend the range of an IoT network when you can't directly communicate from a device to a gateway, is to route across multiple wireless hops. And in the '90s there was a lot of activity on an area known as MANETs, or Mobile Ad Hoc Networks, which looked at the question of how to have multi-hop communication when the devices were moving.

In the late 1990s and early 2000s, research was done in the area of sensor networks, where a lot of the problems that were tackled, fundamental research was done on how you can apply those ideas and come up with new ideas, particularly around low power, to do multi-hop networking between devices. Multi-hop wireless networking between devices at low power. And a variety of papers, a large number of papers were written on this topic.

And in the mid 2000s, mesh networks for the internet were pioneered. And a lot of this research happened at various universities, including MIT, on all of these different fields. So mesh networks for the internet, where you could have devices that don't directly communicate with a router in your home, but go across multiple wireless hops, were again a very active area of research.

Over the past few years, all of these technologies have come together to allow us to build mesh networks for the Internet of Things. So I've shown two examples here. One of these examples is known as Zigbee, which uses multi hop mesh networking technologies to build Internet of Things systems, communicating across multiple wireless hops. The other here shown on the right, is a technology called 6LoWPAN, which stands for IP version 6 over low power personal area networks, which is quite a mouthful. But 6LoWPAN just shows how to build multi-hop mesh technology running over the internet protocols IPV 6 directly, and communicate with the rest of the internet.

Both of these typically run over a low level Mac standard called 802.15.4. I'm mentioning it here because you see this name thrown around a lot. So I think it's just important for you to realize that that is a lower layer on top of which you can put in things like Zigbee or 6LoWPAN. And the use case here, as I mentioned before, our device is communicating with a gateway and then to the rest of the internet across multiple hops.

Now one of the key things about multi-hop mesh networking technologies for IoT is that now some devices, some things are going to do much more work than others. The things that are close to the gateway are going to have a lot more traffic going through them in either direction. And so they're going to do more work, and they might consume more energy. And their battery might last less. So typical rule of thumb here is that people often apply these kinds of technologies when the nodes are powered, or some of the nodes are powered, or where it is in fact, possible to go and change the batteries or have the nodes be powered, say using solar technology.

The other rule of thumb here is that typical experiments and deployments, what people have found is, that some of the nodes could have a duty cycle as high as 1% or 2% of the time, even when the application itself is running a duty cycle, about one in 1,000. In other words, 0.1% of the time the application might have data to send or receive. But some of the nodes might be active, maybe 10 or 20 times more often than that duty cycle. So that's just something to keep in mind when you apply mesh networking technologies like Zigbee or 6LoWPAN to your application.

Now one of the key principles in wireless mesh routing is what paths should packets use in a wireless mesh network? And I want to tell you one example of how to do this. It turns out to be a very interesting example, a very compelling example, and one that used a lot in practice. So let's assume a simple model of the network. It's a very realistic model, but a very simple one where each wireless link delivers a packet with some probability.

Unlike a wire, a wireless links suffer from noise and interference. So they're not very reliable. Some of them are more reliable than others, but they almost always have a packet loss probability.

So let's assume that each link delivers a packet with some probability. And let's also assume, again realistic model, that when you send a packet, or when a sender sends a packet on a hop to the receiver, the receiver acknowledges it. If the sender doesn't get acknowledgement soon after it had transmitted, it will retransmit the packet. And it will retransmit it several times until the packets received, an acknowledgement is received, or it gives up. So with that model, I'm going to show you a picture here.

The first picture on the left, where you have a device sitting at the bottom here. And it can communicate in two ways to the gateway. In the first model, it has 100% packet delivery probability.

This is a fully reliable link here to another device. And that device has 100% delivery probability to the gateway. That's our first option.

The second option is that this device here can communicate directly to the gateway over an unreliable link, which has a 25% packet loss probability. Or put another way, it has a 75% probability of packet delivery. So which of these two methods, which of these two paths must this device choose? Well, if you look at it and you didn't spend a lot of time thinking about it, you might be tempted to pick the first one, because it's completely reliable. It's reliable to 100%, then the second link is reliable at 100%.

Why wouldn't you do that? And in fact, if you look at finding the path with the highest probability of success for transmission, well the first path, the 100% combined with another 100% has a 100% probability of a successful transmission. So you would want to use that.

But remember that what happens here is that the first option here has two transmissions. So you go one hop on the first hop here and the second hop here. So the number of transmissions is two transmissions.

How many transmissions does the second option take? Well it turns out that a simple calculation shows that if the link has a probability of delivery of 75%, what you can do is to say send the packet with probably $3/4$, it reaches the receiver. And with probability $1/4$, I have to send the packet again.

Probability $1/4$ times $1/4$ I have to send it twice and so on. And if you work the calculation out, it turns out that the expected number of transmissions in the second case is $4/3$. But more generally, it's equal to 1 over the probability of a successful transmission.

So now given that I tell you that we've calculated that the first option takes two transmissions, and the second option, the expected number of transmissions is $4/3$, what would you pick? Well, you would probably pick the second option. And the reason why it's a good option is because you can retransmit a packet that didn't get through. And so it is in fact a better option.

If you want a balance between the amount of time the channel is occupied, and your goal of eventually getting the packet through reliably. Looking at a slightly more complicated second example here, let's say that our device here can communicate on the first path at 100% on this link, which is a perfectly reliable link, and 50% on the second link. And you have a second option, which is 75% on each of these two links. Which of these two would you pick?

Well again, if you were to do the calculation with the expected number of transmissions to successfully deliver the packet, what you would find is that on the first path the expected number of transmissions is 1 for the first link, plus 1 over 50%, which is 2 for the second link. And that's 3. And for the second path, it's 1 over 75% for the first link,

which is $4/3$ plus $4/3$ for the second link. And that works out to $8/3$, which is about 2.67. So again, the second option with 2.67 as the expected number of transmissions is better than the first option, which has 3 as the expected number of transmissions.

So the concept here is known as the expected transmission count, or more generally, it's the expected transmission time if the rate of the links is fixed. And on a single link, this is 1 over the probability of successfully delivering a packet and receiving an acknowledgement for that packet. And for any given path, you calculate this for each of the links. And then you minimize it over all of the possible paths. And there are a variety of different routing protocols that have been developed for mesh networks and for IoT mesh networks that apply this concept in order to find good paths to communicate in wireless mesh networks.

Network Connectivity for IoT (Hari Balakrishnan): Video 6

We talk about how to extend communication range with mesh networking technology. But what if we want to even longer range? What if we want city scale, as opposed to just building or campus scale? So we're looking at technologies to achieve miles of connectivity. Now, when the internet is miles away, one principle you could apply to this problem is data muling. And that's actually what we developed as part of the CarTel research project, where the cars themselves were sensor networks. They would gather data from sensors that were embedded in them, either on the car's sensors and computers themselves, or by external devices. And as they drove around, they would store the data until they got to regions where Wi-Fi or other connectivity was available.

Or in fact, as the cars move past each other, they could even communicate the data to each other, and then whenever one of them reached near a Wi-Fi access point, they would deliver the data. So data muling as a principle is a very good way, in which you can extend the communication range to miles even without using wireless technologies that themselves have a mile of range. So in other words, use mobility as a vehicle to get extended communication range. And this is now successfully used in a variety of applications, where you can tolerate delay. So this is an example of something called a delay tolerant network, or a DTN. If you don't need real time communication between your sensors and servers in the cloud, you could actually have the data be stored and be delivered, and connectivity might present itself within several hours, and then once connectivity is presented, the data is delivered.

But what if you want long range and low delay? So there's a new class of technologies that's coming out into the marketplace now, both from the research side and from commercial deployments, known as long range IoT networks. So examples of this are Sigfox, and LoRaWAN, and various cellular IoT proposals. Some of them are proprietary standard, some of them are a little more open. And this is a new area. We don't really have a winner in this space yet, or even a way to know what the best technologies might be. But there's a lot of interesting activity happening here.

The key challenge is achieve a system that's really low power, so devices could run on batteries and last for months, if not years on a small battery. And the trade off is that again, going through the three dimensions, you get long range, you get long battery life but long range IoT networks, but you get very low duty cycles. And so the focus here so far has been on low or really ultra low throughput applications. We're talking about a few bytes per day. So you can achieve a battery lifetime of say even a few years, on small batteries. Some networks, like LoRaWAN, also include localization capabilities, where using information from wireless signals sent from devices and sensors, a base station that might be a couple of kilometers away, can infer the position of an embedded sensor.

And this is important because if you were to have GPS embedded in all those sensors, GPS consumes a lot of energy, it's a power hog, and that defeats the purpose of running long battery life IoT applications. So having localization built into an IoT network technology is often very useful.

We talked about long range IoT networks. But another way in which you could achieve long range and low delay is to use cellular technologies, of course. Now remember, these technologies have high power consumption, so you would apply them only when battery lifetime is not a significant issue. And it does add that although cellular networks give you what you might think of as low delay, they still may incur hundreds of milliseconds of delay going between sensors to your servers, and that delay may still be a concern for a variety of data intensive latency sensitive applications. And I want to explain that to you next.

Network Connectivity for IoT (Hari Balakrishnan): Video 7

Data intensive real time or latency sensitive IOT applications are increasingly becoming common. One canonical example here are continuous recognition applications. So let me explain what that is by example.

Let's imagine a driving application where, as you're driving, computers and sensors in the car are observing the environment, looking at road signs. So for example here, you see the sign of four kilometers per hour written on the road, and you might have a sign that shows a right turn only on that road. What the system's doing is capturing that image, recognizing that image, and then showing it to the user as an overlay, telling them to slow down if they are going too fast.

Another example of such an application is a user taking their phone and walking around the store, perhaps in a foreign country, and training the camera to look at different objects there and have it automatically translate what the objects are by recognizing the objects and translating the words and the different objects that are seen to something the user can actually understand. Other examples are when you're walking around and you like your cameras, perhaps you're wearing glasses with cameras on them, and you are providing augmented reality, over layers of things in the real world, by connecting information that's observed from your sensors and cameras, translating them or recognizing them in real time using recognition technologies and displaying information to the user.

So Glimpse is a system that we've developed at MIT that does this for users. Glimpse is a continuous real time object recognition system that runs on mobile devices. It captures video at a high frame rate, 24 to 30 frames a second, and continuously identifies and continuously locates objects in a video stream for a user on their mobile device. So examples are shown at the bottom here. This applies it to face recognition where, in real time, at this high frame rate, the application shows not only where the people are, but also who they are to the user.

We've used Glimpse not only to look at identifying people, but perhaps more interestingly, identifying road signs as you're driving. So this technology works in three stages. It first captures images as a video stream, and it first runs a detection step on it. And in the detection step, the job is to figure out where in the video stream and where HMH objects are. The second task is to take the objects of interest and break it down into individual features of the objects so you can then recognize what they are. And the third stage is to classify the objection into one of several kinds.

So in this example here, we go all the way from capturing images on the road to a stop sign, and we'd like to do that really quickly, within maybe a couple of milliseconds. But in fact, run, as far as the user is concerned, at 30 frames a second, which means that we would like to recognize everything on a frame within 30 milliseconds. That's roughly 30 frames a second. This is a challenging problem because this task is computationally very expensive. It takes a lot of effort to extract features and classify an image into one of several possible categories. And servers doing this are hundreds of times faster than

doing it on a mobile device itself. It's actually very, very difficult to do this on a mobile device, even with hardware technologies available on a mobile to do certain kinds of object detection.

The other problem is that scaling to a very large number of images, which you might have in many applications, you might want to scale to say hundreds of thousands or millions of images to classify them, requires servers to run. So the problem is we have a cellular network technologies, or even Wi-Fi for that matter, that will take hundreds of milliseconds to capture images, send them to the server, have them do the recognition, and give answers back. This process takes hundreds of milliseconds or seconds, and what we'd like to do is to provide to the user something that runs at the rate of 30 frames a second. This is the challenge.

So the first step of the solution is to offload computation to the servers, because we just can't do it on the mobile device. The problem is we have end-to-end latency that, if you did everything on the server, would lower the object recognition accuracy. It would cause you to recognize an object, but by the time you tell the user that the object is a road stop sign and it's located at a certain part of the image, the image has shifted. So now you're always inaccurate and falling behind. The second challenge is that if you're not careful, if you send all of the video to the server, you're going to consume an enormous amount of bandwidth and consume a lot of battery.

So how do we achieve this goal of providing real time response when the control loop between capturing an image, delivering it to the server, recognizing it, and bringing it back may take hundreds of milliseconds, or a couple seconds. The key idea in how we're going to provide real time object recognition is to use a device-side active cache. Let me show you how the active cache works with an example.

We're going to number our frames from 0, 1, 2, 3 all the way forward. Frames have been captured full motion video at 30 frames a second. So let's take frame zero and send it to the server, because the server has to do the hard work of recognizing the different parts of the image. So while the server is doing this work, the device is moving forward and capturing subsequent images. When the server finally returns a response to us, that response is going to be several frames old. But what the active cache allows us to do is to use that old response and quickly, in real time, run the tracking of that labeled object through the frames that are stored in the active cache. So as far as the user is concerned, it's going to have the correct label that's shown in the image in the view of the user.

So this is the key idea behind the active cache, and once we build up this pipeline-- our first frame is going to be delayed, but then we're going to catch up in real time, and then moving forward, we're going to be able to not only capture the videos in real time, but also label and recognize all of the objects in the video in real time.

So another way to think about this concept is that the server, from time to time, is going to give us information about the different objects in the image. The majority of the work is being done by the server, but the real time responsiveness is being done on the mobile

client itself, because what the mobile client does is it's able to do the tracking of labeled objects in real time moving forward to a video stream.

Here's what happens if you didn't use the active cache and you sent images to the server and have them do the recognition and then come back. You can see that because of this delay, the images aren't tracking at all. The objects are not being properly recognized and located in the image, and that's the problem we'd like to correct. With the active cache and the method that I described, you can see that what happens is that the objects are being located and tracked in real time, and that's because we have the mobile device being able to run part of the computation locally and only have the hard part of the computation, namely the feature extraction and the actual object recognition, run on the server side. Once an object has been labeled by the server, the rest of the tracking happens in real time on the mobile device itself. It's only when new objects show up that we have a little bit of delay and then, after that, we can track locally.

So how does this work? Well, we can think of performance in terms of the precision and the recall of object recognition. So precision is the number of images that we correctly identified and labeled, divided by the total number that we reported. And of course, we might report objects where none exist, and that's a bad thing. And recall is the total number that we correctly reported, divided by the actual number of objects in the [INAUDIBLE]. So these are two different axes along which we measure accuracy. And the third thing is we measure the bandwidth consumption.

Now for faces, if we were to use the server alone and we send a frame, have the server recognize it, and come back and do the same thing again, you can see that the accuracy is around the 50% mark. And with our technologies, with the active cache, what you find is that we are north of 90% to 95% in terms of the overall accuracy. It's a pretty big win.

And one of the other things we can do, I didn't talk about this, is that we don't have to have one frame outstanding at a time being recognized by the server. We can actually send frames only when the scene changes substantially. We call those triggered frames, because the rest of the time of the image is more or less the same, and people are just moving around or objects are moving around in the same image. You don't actually have to burden the server. And if you do that, the bandwidth consumption drops by a dramatic amount, but the accuracy remains roughly the same.

The performance of Glimpse on road signs is even better. With server only, you end up with very, very low performance. And one of the reasons for that is that in the previous results, we actually used some hardware capabilities for face detection that's available on mobile devices, but there is no such capability available for road signs today. But if you run Glimpse, what you find is that the accuracy on road signs in terms of precision is about 80% and it's about 60% on recall, which is pretty good for many automotive applications and certainly a lot better than if you didn't have it. And again, if you were to reduce the bandwidth usage by only sending frames where substantial difference has been observed, you end up not changing the accuracy a whole lot, but the bandwidth consumption goes down a great deal.

Network Connectivity for IoT (Hari Balakrishnan): Video 8

This brings us to the conclusion of this module. So what have we learned? First, there's a rich design space for communication between things and gateways. That even though there are a very large number of design choices out there, fundamentally, this entire topic boils down to different ways in which you can communicate between the things-- between sensors, and actuators, and devices-- and the gateway. Because once you get to the gateway, that does the job of translating and mediating to get to the rest of the internet, and to the servers in the cloud.

Second, the way to cut through the complexity of the connectivity soup of IoT is to think along three dimensions. Number one-- duty cycle and data rate for your applications. Number two-- the battery lifetime of the system. Number three-- the communication range.

Third, we looked at three case studies. The first of those case studies was low-power design Bluetooth Low Energy. We talked about how advertisements worked, and the time-scheduled MAC protocol, and went through a calculation that showed why BLE is able to achieve about three years of battery lifetime on a small console battery.

Second, we looked at different ways to achieve range extension. We talked about data muling, which is to use mobile devices, and people or automobiles, as they move around, until they get network connectivity to deliver the data. So you trade off delay for communication range. So you can get miles of connectivity without requiring a network technology that will give you, now, miles of connectivity. You can rely on mobility to extend your range.

And second, we talked about mesh networking, in particular, Zigbee and 6LoWPAN. And third, we talked about data-intensive IoT. We went through some of the challenges and techniques that we use to build a continuous object-recognition system.

This is a relatively new area, and there's a very large number of open questions and future work. To paraphrase Winston Churchill badly, we're not even at the end of the beginning of IoT. One example for why we're not even at the end of the beginning of IoT is these applications that you see now, where people are building for IoT, where you have to build an application on your smartphone to turn on the lights in your room. This is silly. Who's going to enter into a dark room, pull out their smartphone, get to the right app, click on the thing for on, and do the same thing when they go off? That doesn't seem like an exciting IoT application. But, in some ways, those are the kinds of applications that we're looking at today.

And what if you want city-scale, high-rate, low-powered sensing? Examples of those include high-fidelity vibration sensors, for example, monitoring earthquakes, or weather sensors that high fidelity, or image sensors. We don't really have a good answer to this question today, if you want all three of these.

Third, current systems are often gated by standby power, which is in the microamps of current draw multiplied by about three volts of voltage. It's microwatts of power. Recent advances in hardware groups at MIT have shown how to cut the standby power down from microwatts to nanowatts. This technical advance will make significant changes to the design of many low-power IoT networks.

And the last thing I want to say, in terms of open questions and future work, is that current Internet of Things applications are siloed from each other. I'll explain what I mean by that. But the real open question here is, how do you really integrate data from different applications?

So today, the way in which you go about building an IoT system is to take IoT devices or sensors, or build them yourself. Then you go ahead, build an application, typically on a smartphone, or you might have an application running on a gateway. And then you build a cloud service to process the data once it reaches your servers, in order to derive inferences from that data.

These applications are highly vertically integrated. You have to build everything, from the devices and the sensors, to the communication, to the gateway, to the cloud service. And the result is that it's very hard to integrate multiple IoT applications together. And it really slows innovation.

The other point here is that gateway functions, the task of simply getting the data in and out of the sensors, is repeatedly invented, over and over and over again. This is very different from the internet. The issue for IoT is that the real value in IoT systems is in the data, not in the devices. And not even in just achieving connectivity between machines, or between a device and the internet. The value is in the data.

This is a little bit different from on the internet where, fundamentally, access to machines was the killer application. So you could build a wonderful internet architecture and succeed, and allow for applications to be written separately from the networking protocols, and have a wonderful, layered structure. And it was successful, because the value wasn't achieving connectivity. But value and IoT is often in the data. Simply being able to connect to a device is just not interesting. What you do with the data is where the value is. And that's why we've ended up in this highly-siloed architecture.

Who wants to simply build network technology for IoT, when the real value, and the money to be made, is in processing the data? And this is why we see a very significant degree of vertical integration here.

So what are some solutions here? So here are some possible non-exclusive approaches to the problem. That is, you could combine these things together. First, a good approach might be to coordinate access to the data via server-side APIs in the cloud. In other words, it may be, for the foreseeable future, in many applications, that we are vertically integrated. That you have your home thermostat application, and you have your automotive application, and you have your fitness application. They're all completely,

vertically separate from each other. But the integration happens with cloud-side APIs between servers. Then I think that's the way in which you see advances being made already today.

Access to data in smartphone applications via "kits." So examples of this are Apple's HomeKit and HealthKit and Google Fit and so on. Now there are trade-offs in doing this. One of the trade-offs is, of course, you now buy into a particular smartphone ecosystem, so it becomes hard to have devices in one ecosystem communicate with another. And also, it presumes that the smartphone is at the center of all of this activity, which is only true for a fraction of IoT applications.

And third, perhaps somebody will, in fact, find it useful to develop generic gateways. And that gateway has to include multiple technologies. This is tricky, because it's hard right now to predict the winning technologies, and you might end up with multiple different kinds of gateways. But perhaps it is still valuable to think about the right software interfaces to extract the data out, so that people can write applications around the gateways and not worry about the low-level details of how to get data in and out of the devices.

To conclude, here are five predictions of the future of network connectivity for IoT. In my opinion, there's going to be a shake-up in standards. We will see multiple winners. There's not going to be a single winner taking everything. And that's because one size does not fit all. But I believe that the winners will divide up the three-dimensional space that I showed, and take pieces of that for themselves.

Second, we will see a lot of excitement in ultra-low power IoT systems. These are of the Nano watt level of standby power, and the network architectures that go around it. It's not going to be all of the IoT space. We're going to see a lot of IoT running with the techniques that we talked about. But I think ultra-low power systems are going to become exciting.

Third, compute-intensive IoT systems are going to become exciting. Today, many IoT applications are not compute-intensive. You know, you get data out of your heart-rate monitor, and your fitness, you do some processing in the cloud, but it's not really data-intensive. But I showed you an example of continuous recognition as a canonical example of where it's data-intensive and compute-intensive. And I think a significant fraction of IoT is going to be in that high-rate, data-rich system.

Fourth, de-siloed architectures will happen. It's already starting to happen with cloud-level APIs, and via these "kits." Advances in cloud-to-cloud APIs for different IoT applications is going to be an important area for the future.

And last but not least, we're going to see an interesting segmentation between smartphone-centric IoT systems, and hidden, truly ubiquitous, computing like IoT systems. And a large amount of the IoT activity today, the ones that are really popular, are, in fact, smartphone-centric, where the smartphone acts as the gateway. And it really

isn't hidden. The smartphone is front and center. It's the user interface, but it's also the gateway technology that's used to connect to IoT.

And that will remain important. But there's a different class of IoT systems, like for factory automation, and home automation, and other such applications, we're seeing significant effort in. We will see that evolve into widespread adoption. And it will be interesting to see how these two combine, if they do, and what the future for IoT that combines both smartphone-centric and hidden technologies looks like.

So thank you very much for your attention. It was a pleasure bringing this lecture to you today. And I hope you enjoyed it.

Data Processing and Storage (Sam Madden): Video 1 Part 1

Hi, I'm Sam Madden. I'm going to talk to you today about some data management issues that arise in the Internet of Things. Before I start telling you about some of the data management challenges, I first want to talk about my own personal experience developing and deploying Internet of Things applications.

So I'm a professor at MIT, and I've been here for about 10 years. And my research area is building systems and software systems to manage really large amounts of data or complex data from things like little embedded devices such as a rise in the Internet of Things world. So I'll just tell you about a few of the applications that I built myself.

As a graduate student, I worked on a project called TinyOS where we built an operating system for these tiny embedded battery-powered devices that were connected together over large wireless multi-hoc networks of the sort that you heard about in a previous module. I also built a system called TinyDB where you could treat a collection of these little devices as though they were a database system that you could interact with in a language like SQL, this sort of Esperanto of databases. So you could really sit it down at your terminal and run a query that asked the network of these things to produce some sort of answer for you.

When I got to MIT, we started developing an application called CarTel. This is as in car telecommunications not drug cartel. We built an app for iOS called iCarTel, which was a crowdsourced traffic application. This was before things like Waze and Google Maps existed that could give you real-time traffic data. And it allowed users running our application to see traffic on a real-time display and get traffic aware routes.

More recently, I built an app to control home lighting system built by a company called Lutron. And I've been heavily involved in a local startup company that's built an app called DriveWell that uses a little embedded device, a Bluetooth-powered accelerometer, to capture information as users drive, in order to provide them with feedback about whether they are safe drivers or not and also to do real-time collision detection and alerting. And this little tag device relays information through a smartphone app and into a cloud where users' data is processed.

All right, so with that introduction to my own personal experience, what I thought I would do is just talk a little bit about the need for data and Internet of Things. So some of you may be wondering why is data important in the Internet of Things? But if you start to think about a lot of the IoT applications, you actually realize that data is at the core of them.

So consider applications in the space of infrastructure monitoring, like home monitoring, or monitoring pipes or other industrial equipment, or medical device monitoring. These are really about understanding when something interesting happens in these monitored devices. And that interesting thing that happens is really fundamentally conveyed in data.

For example, you might want to know that the temperature in your home went below some threshold, and the pipes are about to burst. Or with a medical patient, you might want to see some signal, like a brain or heart signal, that is showing some sort of anomalous value. Lots of other applications have this kind of nature as well.

So if you consider vehicle tracking, you're interested in the locations of the vehicles, perhaps, what the drivers are doing. If you think about home security, you're interested in understanding when somebody is breaking into your house, maybe seeing pictures of that. So all of these applications have really data at the core of them that has to be brought back to some user and displayed.

So just to give you a sense of the typical Internet of Things data processing architecture, I just want to walk you through how many of these applications work in the sense of how data flows through them. And I think this architecture is one that you've seen in previous sections. But I just want to overlay how the flow of data works here.

So many of these applications really consist of a collection of sensors that are deployed in the world. And they're typically relaying information or data through some sort of a base station or maybe a smartphone. And then that data is typically delivered to a server, which is running out on the internet somewhere where much of the processing will happen.

So the path that data will follow in this is starts on the sensors, it flows through the phones and base stations, and then ultimately ends up in this cloud-based infrastructure. And then in many of these deployments, especially ones that involve smartphones or other end user display devices, you will send the processed information about what happened or the interesting things that were in the data back to the users. So looking at this architecture, there are just a few things that I want to point out.

First of all, these sensors are typically connected through these low power wireless communication protocols of the sort that Hari Balakrishnan talked about in the previous module. These are protocols like Bluetooth LE or Zigbee. The phones and base stations themselves are frequently talking over WiFi or some other maybe wired connection to the internet. And then, the data, the cloud, is residing on the internet somewhere probably remote from the actual location that is being monitored.

And in this architecture, there is data processing that can happen at all stages. The sensors may do a little bit of processing of the data. The phones or base stations may filter it--may discard uninteresting events. And then, ultimately, the cloud's infrastructure is what's going to be aggregating data from many different devices together and often doing the most interesting parts of the processing.

Given that sort of introduction and canonical architecture for what an IoT data processing application might look like, I want to talk a little bit about some of the challenges in doing this. Why is this hard or different from the kinds of data processing applications that you might be familiar with, like database systems or building web applications? IoT

data is really fundamentally different than the kinds of data that you encounter elsewhere on the internet.

And the reason for this is that the data in IoT is being produced by sensors, things that are deployed in the physical world. And those sensors are really sampling the physical world. They're not observing everything. They're observing a sequence of discrete time points.

Imagine, for example, a camera that takes pictures whenever it sees motion in front of it. So you don't have every picture in the world. You only have a subset of the pictures that capture some subset of the interesting things that happen. And all these other sensors that are deployed in Internet of Things have a similar property.

Accelerometers, for example, that are measuring vibration, are only measuring vibrations up to some maximum frequency. Gyroscopes that measure the movement say of a phone, again, have a limited resolution and precision.

The other thing that makes these IoT data applications challenging is that many of these devices that are deployed in the real world experience noise. To give you an example of what I mean by noise, consider a digital photograph. So your digital camera is taking a picture of the real world. It is a sensor.

And many of you have, perhaps, tried to take a picture in a low light setting and seen that the picture is blurry or smeared or appears to have little artifacts in it. Those are examples of noise that are-- because the sensor is just not quite good enough to capture a perfect image in a low light setting. The same types of noise issues arise with all types of sensors that are deployed in Internet of Things applications. And I'll show you a few examples of those in a minute.

So just to summarize the key challenges that IoT applications have to deal with, they are first of all, limited resources. These sensors are deployed in remote locations. Oftentimes, they're battery-powered. And they're subject to the limitations of the networks that have to be used to deliver data back from them.

Second, this issue of missing and noisy data. There's noise in the data. And oftentimes, these sensors, because they are sampling the real world, have periods of time that are not covered by the data itself.

Finally, the data that is coming from these sensors and these applications often has anomalies in it, things that are unusual or outliers. And so one of the real challenges that we have to deal with is how do we detect and correct those kinds of outliers and anomalies. So these three key issues are the ones that I want to spend the rest of this module describing to you.

Data Processing and Storage (Sam Madden): Video 1 Part 2

Before I move on to the technical details of these issues, I just want to give you a reference picture, a picture of how one of these applications that I've been involved in works, because I want to refer to it several times throughout the rest of the module. So this is this case study of the DriveWell application that I mentioned at the beginning. You remember this is an application that gives users a safe driving score-- it tells them whether they're driving well and it points out locations where maybe they've done unsafe things, like braked very abruptly or swerved. It also, through this little tag device, provides collision detection facility. This tag is fixed to the inside of the car, and if the vehicle experiences an accident, the application running on the smartphone can immediately relay an emergency alert up to an emergency service providers.

Here's an architecture of how this application works. There's acceleration data that's being captured by this tag device. This acceleration data is used to detect impacts-- it's also used to detect when the vehicle starts moving. This data is relayed over Bluetooth, low power Bluetooth, to a smartphone. And the smartphone itself collects some additional information, for example, information from a gyroscope on the phone to understand whether the user is using the phone while they're driving, which might be an unsafe kind of behavior. This data then gets relayed to this Amazon AWS Cloud, where the data is processed, and many events are detected, in this driver score is produced.

So there's a couple of interesting requirements in this application that I think really motivate why you need to be really careful about data processing in an internet of things setting. So the first one is a requirement that this tag last for multiple years. So this tag device has a little battery in it. It's fixed to the inside of the car, and we wanted it to last for something like three to five years without needing to be replaced. So this imposes very strict constraints on how much data we can send and how frequently the device is active.

Second of all, we wanted the phone itself to not use too much battery, because users are going to be very sensitive to an application which drains the battery on their device very quickly. Third, we have end-to-end latency requirements. So for example, when a collision is detected that information, needs to be relayed to the cloud so that an emergency service provider can be contacted within just a few seconds. So there's a very strict real-time latency bound on this collision detection feature.

Although there's a less strict time requirement for other types of data that this application collects, it's still important to get relatively timely responses for other aspects of this application. So for example, when the user completes a trip, they might like to know within, say, a few minutes, how that trip affected their score or whether they had any severe events detected in that score. And then finally, it's important to get some level of accuracy out of this application. So this application is collecting information as people drive, and it's giving them some feedback about where they had a harsh braking event at some location or whether they were speeding on a particular road. And users don't like it

when the application gets these things wrong. So it's very important that the application actually accurately measure what happened in the real world.

And to give you an illustration for why this is a challenge, I just want to show you a couple of pictures-- and I'll come back to both of these pictures in a later portion of this module. So the first picture that we're seeing here is just a picture of a GPS trace captured from a real device. And so you can see approximately where this person drove here, this green line, but you can see that the green line really doesn't follow any roads perfectly well. In particular, you can see at this location here that the GPS shows the user driving off into somebody's house, which he almost certainly didn't do. And you can see several other locations where there are weird outliers or anomalies in this GPS data. And that's something that we're going to need to correct in order to tell the user what roads they were driving on and whether they were speeding. If we told the user that they drove into somebody's front yard, they would immediately discredit, and probably stop using, our application.

A second kind of issue that arises with the data in these applications is that the data is very noisy. So here's just a very simple example of noisy data. You see this big jostling in this accelerometer data here. What this represents is the user, as they were driving, picked up the phone and moved it. And if we weren't careful, we might think that this jostling represented the user doing something very unsafe in the car, whereas in fact what really happened is the user just picked up their phone, say, and shifted it from one pocket to the other or flipped it over to look at a text message or something. And so we need to make sure that we don't confuse these kinds of noisy events in the data with real events that are affecting the dynamics of the car on the road.

So the third challenge that arises with data and outliers and anomalies in these kinds of applications is that in the real world when you're deploying applications, especially on things like smartphones that are this very complicated ecosystem of software, you often experience unexpected behavior in certain environments or settings. So this issue that I've shown here, there's a Discover CBCharacteristic for CBSERVICE. This is a bug that arose when Apple released a new version of their operating system, and it caused a large fraction of the devices that we had deployed on the roads to simply stop being able to record data. So it's very important to detect these kinds of anomalies very quickly. As soon as this new operating system goes live and is in the field we need that information. So we need this real-time anomaly detection, not only on the sensor data but on other aspects of the data that is coming from these devices in the field.

In the rest of the module, I'm going to focus on the three key challenges that I outlined a couple of slides ago. The first one is the issue of limited resources-- how do we deal with the fact that these devices that are deployed in the real world have unlimited power and bandwidth? Second of all, I want to talk about this issue of missing and noisy data, the fact that these signals being captured from these sensors have lots of noise in them, sometimes they have gaps, they have other outages that need to be corrected and cleaned. And then third I want to talk about this issue of outliers and anomalies-- how do we detect things like the fact that a particular class of devices has stopped functioning in

some particular environment, as in the example of Apple releasing a new version of the operating system.

Data Processing and Storage (Sam Madden): Video 2 Part 1

In this segment, we're going to talk about the issue of handling the limited resources that internet of things devices have. By resources I mean things like network bandwidth or battery power. So before I start talking specifically about these resources, I want to sort of point out a challenge that many builders of internet of things applications have, which is that there's a natural tension for people who are processing the data. The people who receive data are trying to make sense out of it, to want to capture as much information as possible at the highest possible fidelity.

In contrast, the people who are building these application platforms, the devices and sensors that go out in the world, have to confront the reality of the resource constraints that they have. So they have a certain size battery or a certain amount of network bandwidth that they can use. And this is going to naturally limit the amount of data that they're going to be able to collect.

So some examples of constraints that an application designer might face are that the battery powered device needs to last at least x years. And there's a maximum number of millimeters in the package of the device that can hold the battery. Or the radio can only transmit at y bytes per second. And that's going to mean that there's only a certain number of data readings per second that can be collected. And third, this device can say, for example, only store a limited amount of data. You have an audio or video recording application. And it's going to be limited by the onboard storage and the cost and the amount of money that you're going to spend on putting storage on your device.

All right. So with these constraints in mind, I want to talk about the three key ones that I think arise. I'll call these the three Bs. So these are battery, bandwidth and bytes. I think the real main ones to focus on here are battery and bandwidth. But I'll talk quickly about bytes, the storage issue, at the end.

So really in battery powered applications it's often going to be the fundamental constraint that you have to deal with. It just limits how much you can collect. And in these application scenarios there are certain technologies that can really dramatically affect the overall battery life. So for example, if you want to add a cellular radio to a battery powered device, that's going to use a ton of the battery on the device itself.

To get you in the mind of thinking about how much battery is going to be used by different components and the overall battery consumption of an application, I just want to do a very quick recap of electromagnetism. So you guys probably remember that joules are the standard unit of energy. OK? And the standard unit of power is a watt. So a watt is the amount of energy that you burn, the number of joules that you burn per second when you're running some device.

So if I turn on the screen on my smartphone, that's going to use a certain amount of power. It's going to consume a certain amount of the capacity in the battery every second that the screen is on. So the wattage of a device determines how quickly it drains power.

And wattage is just the amperage times the voltage. So you guys probably have seen that a typical battery powered device might run at 3 or 5 volts. And it would drain some number of milliwatts.

So the wattage determines the power consumption, the power drain, of these devices. And the battery capacity is its stored energy. And that's typically measured in what's known as milliamp-hours. So to give you an example of how we can use these numbers to think about how much energy a particular component of an internet of things application is going to consume, I'll just give you an example here using an iPhone 6.

So an iPhone 6 has an approximately 800 milliamp-hour battery. So that means that it can run at 1 milliamp for 1,800 hours, or 1,000 millamps at 1.8 hours. So an LTE radio, the most current generation of cellular radio on these devices, uses about 7,800 milliwatts when it's transmitting at 1 megabit per second.

So in order to figure out how long we could run the radio on an iPhone 6 continuously for, we can just do a little bit of simple multiplication. So the iPhone runs at 3.8 volts. So 1,800 milliamp-hours is the same thing as 6,840 milliwatt hours at 3.8 volts, because watts is just amperage times voltage. So we get 6,840 milliwatt hours divided by 1,700 milliwatts to run the LTE radio. And we find that we get about four hours of continuous battery life out of an iPhone 6 if you were to run the radio without interruption for that entire period of four hours.

So this LTE radio uses a lot of energy. It's probably the most power consumptive part of the phone. And so in cell phone based internet of things applications it's really important to not turn the radio on and leave it on for long periods of time. So just to give you some rough sense of these different types of numbers if you're curious about them-- I won't go into all the detail-- but you can see that these LTE and 3G radios both use around 1,700 milliwatts.

Wi-Fi uses much, much less-- about a quarter of the power consumption of these LTE radios. So if you can use Wi-Fi you can save a lot of battery life. The microprocessor also uses a lot of energy. So if I run the processor on these devices at the highest 100% rate full out doing some really expensive computation, it uses about 2,000 milliwatts. But when it's idle it doesn't use very much. So again, you can do some computation. But if you continuously compute on these devices the battery will drain very, very quickly.

Some of you have probably experienced this. You're running some game or something that's really running the phone full out. It starts to heat up in your hand. That's a good sign that it's running the processor really hard. And you see that the battery drains very quickly when you do that. On the other, hand, if you divide that 1,800 milliamp-hour number that I talked about before by 70 you get some number that's around 24 hours, which is sort of about how long you'd expect a smartphone to run if it was just idle in your pocket and you weren't really doing anything with it.

The screen at full brightness also uses quite a lot of power. So interestingly, the last three things here are sensors. So the sensors on these devices use some power, but frequently running the sensors actually isn't the most power consumptive part of this. So a GPS it can sometimes use large amounts of energy. But modern GPSs are actually quite good. They only use something like 100 to 150 milliwatts. So like a factor of 10 less than running the radio.

More surprisingly perhaps, if you have a video sensor. So something that's capturing 1080p, 30 Hertz a video-- really a large amount of data and capturing at a very high data rate. This thing only uses 270 milliwatts. So this just gives you a sense of the key things that you really need to worry about in these data applications are first of all, radio. Because it uses so much energy. And second of all, being careful with the microprocessor.

So there are a few caveats with these numbers that I've just given you. First of all, all of those numbers are approximate. They vary quite a lot from one platform or one generation device to the next. So don't be too literal about those numbers. If you look at a different platform you might find something that's different by 25% or more. Second of all, there are some other issues or subtleties that arise when using these kinds of devices.

So the first one is start up and shut down times. You might say I'm only sending a few little messages over the radio. So I'm not really using very much data. And so I'm not going to be running the radio very quickly. The challenge with that is that if you're sending a few little messages but you're sending a message every second, it actually turns out that typical phone radios will not actually shut the radio off during those periods.

So there's typically like a 10 second shut down period on these radios, which means that if your application sends a message, waits three seconds, sends another message, waits three seconds, sends another message and does that over and over again, the radio will never shut down. So you actually really want to be careful in these applications to remember to try and space out the deliveries of data, even if you're sending relatively small amounts of data.

The second issue has to do with the power utilization versus processing efficiency. So you might have looked at those CPU numbers and said well the CPU running at 100% that really uses a lot of power. So maybe I should use a lower power CPU or I should run the CPU at less than 100%. But the thing to remember is, of course, a faster, more powerful CPU can get your computation done more frequently. So it's not always obvious that just going to a lower power CPU is the right solution in these kinds of deployments.

The third issue is similar to the previous issue with CPUs. It has to do with radio power versus bandwidth. So when you do have lots of data to send, even though these radios use a lot of energy, if you use a higher bandwidth radio that can transmit faster you might be able to use a lower total amount of energy, even though the amount of energy per second - the milliwattage-- of these higher power radios can be a lot higher.

And then the fourth and final issue that arises with radios is that you have to be a little bit careful, especially these 3G radios have an asymmetry between the power consumption of uploads versus downloads. And that can be pretty significant like almost a factor of two.

Data Processing and Storage (Sam Madden): Video 2 Part 2

I just want to talk a little bit about how these power considerations affect IoT applications. So first of all, as you can see from this previous description, the more data you have, the more data you're going to have to send. And that's going to cause you to use more energy. So I know that's sort of an obvious observation, but when you're building an application deployment, remember, your data scientists are always going to want you to collect more data. So you have to think really carefully about how you only collect exactly the data that you need.

The other sort of consideration that you have to take into account is whether you want to favor on device processing, which is going to use more CPU but reduce the amount of data transmission, or whether you want to just send all the data off and process it in the Cloud. And so consider a video processing application. A smartphone or an other embedded device might have enough CPU power to do some of the processing locally. The question is whether that processing, which may be quite intensive for a video processing setting, is better done there, or whether you would actually save power by uploading the video to the Cloud and processing it there.

And then, the third consideration to just bear in mind is that, remember Wi-Fi typically uses a lot less energy than these 3G and 4G radios that are on cell phones. And so, if you have a cell phone based deployment, using Wi-Fi to get data off is typically going to be much, much better. So if you can wait until you have a Wi-Fi connection instead of sending data immediately that can be a big savings, in terms of the data that you collect.

In order to illustrate just one of these, what I think is kind of an interesting example about reducing the amount of data that you collect and transmit, I want to talk about an application that I built, actually quite a long time ago, as a part of my Ph.D. Thesis. So this is this TinyDB application. Remember, I mentioned in the introduction that TinyDB is an application where you could treat a network of devices as though they were database that you can run a query against. OK. And so what TinyDB did is it used something called Multihop data collection. So you have this sort of mass of devices, little sensors, deployed out in the world. So we did things like put these in a Redwood forest in the trees in Berkeley, or monitored a population of little birds off the coast of Maine. We used this for a number of things like this. And in these settings, the radios, the devices, organize themselves into a Multihop configuration, where they relay data from the very remote nodes, back up to some root node which is connected to the internet or some other base station, in order to get the data back. So you can sort of think of these devices as being arranged in a topology that looks like this, where this sensor 5 is the furthest away from the base station, and sensor one is the one that's immediately connected to the internet. So this is sometimes called Multihop data collection.

And the idea in TinyDB is that what we wanted to do was to make this data collection both energy efficient, as well as preserve and limit the amount of bandwidth that's consumed. So, consider a very simple query. So suppose you're running the simplest possible thing, which is just to count the number of devices that are out there. So in

TinyDB to be could run this query. You could say, `SELECT COUNT(*) FROM sensors`, and it would produce an answer. In this case, 5. So to show you how this works, consider how data would flow through this tree. So the very first step here is that sensor ID 5 would transmit a value of 1 because it just counts itself. It sends that value of 1 up to its neighbor, its parent in the tree, which is node 4. Node 4 receives that value of 1, and adds its own count to it, which in this case is 2. And this process just continues up the tree, until eventually we see that all of the sensors accumulate their count. Sensor 5 transmitted a value of 1, sensor 4 transmitted a value of 2, sensor 3 transmitted a value of 3, sensor 2 transmitted a value of 1, and then finally sensor 1 transmits this value of 5, indicating the overall count.

So this is a trivial query, but it can be generalized to apply to a very large class of the kinds of things that people want to do with these kinds of sensors. Imagine you want to measure the maximum temperature in some kind of a building deployment. That's something that you can do very naturally with a query like this. So, the cool thing about this particular way of structuring this is that notice that every sensor in the setting has to transmit only one value, the count of itself plus its children. And also notice that because of the way that we've constructed this, if you think about there being time intervals in which these sensors transmit, where time is divided up into these slots, the sensors actually can sleep during most of the time. They only need to be awake and listening and have their radio powered on for these intervals when they expect their children to transmit.

So we used careful time synchronization between these devices, and this sort of clever way of aggregating the data as it flows through the network, in order to minimize the amount of time these devices are on, listening to their radios, and the total amount of data that they have to transmit. So there are some kind of clever and interesting things you can do in many IoT applications, when you think carefully about how to structure the nature of the communication, and synchronize the times when the devices are operating.

So the second issue that you have to deal with in these Internet of Things applications is bandwidth. The bandwidth issue arises because the radios that are on these devices have a limited number of bytes that they can transmit per second. Consider, for example, the Bluetooth LE radios, that many small Internet of Things devices may use to connect to a base station or a smartphone these radios can only transmit a few tens or hundreds of kilobytes of data per second, and that's really going to limit how many readings you can get off of these devices. Even in some higher power radio settings, like in a cellular phone with a 3G radio or an LTE radio, you may not be able to transmit all of the data, if you're doing some particularly data intensive application. For example, if you're trying to stream high-rate video off of one of these devices.

So the previous module, where we talked about networking technologies in Internet of Things talked about a lot of these issues in much more detail. But there are a couple of considerations that I think are worth mentioning when we talk about data processing in the Internet of Things. The first of these is this issue of continuous monitoring versus alerting. Whether you're trying to just stream data off at some background rate, or

whether you want to send some immediate alert up to a user, like in our DriveWell application where we needed to alert people of collisions. The second issue is that it's often important in these settings to employ techniques like buffering, in order to be able to smooth over various kinds of disconnectivity or intermittent connectivity that may arise. Because, as we're all aware, these radios on our cell phones, for example, don't necessarily provide continuous connectivity. We may be in some area, or there may be some temporary interference that makes it so that we can't transmit data at a particular point in time.

In order to handle this issue of intermittency or dropouts, as I mentioned, the key thing to do is to really employ some form of local buffering of the data. So you store the data locally, on hopefully the sensor, and if there is some bay station on the bay station. And then, once you're sure that it's been properly delivered to the end point, up into the Cloud where it's been safely stored, then you remove the data. And you need this sort of round trip acknowledgement procedure where you'll detect that the data has actually arrived at the bay station. And only then will you remove it. So this kind of looks as follows. If you've got a sensor with some data on it, it's going to transmit that data through some bay station, like a phone, and then up into the Cloud. So this data first goes to the phone. The phone stores a copy of it. Then the data goes the Cloud. The Cloud stores a copy of it. And only then is it OK to remove the data from the phones and the sensors. So once the Cloud sends back an acknowledgment, then you can go ahead and remove this data from these sensors and phones that are out there in the world. So we call this sort of an end-to-end principle. You really should only remove the data when you're sure that the end point, the application on the other end, actually has it.

So just to give you a quick illustration of how some of these issues around bandwidth arise in real applications, I want to return to our example of the DriveWell plus the TAG application that I mentioned in the introduction. So remember, this application collects driving metrics. These amount to hundreds of kilobytes of data per hour. And it also does this real time impact alerting. These impact alerts are relatively small, they're just a few tens of bytes to transmit the information about the impact. So what we do in this application is we actually buffer these metrics, the driving metrics, locally onto the phones. We store them in files, and then we upload those files when Wi-Fi becomes available, because Wi-Fi uses so much less energy, and also because sending hundreds of kilobytes of data over a consumer's cellular data plan may be undesirable. This data, this driving metric data, is only removed from the smartphones when we get an acknowledgement from the Cloud server that, not only with the data received, but it was actually properly processed and there were no issues with it.

Second of all, we relay these impact immediately using the 3G radios. These are small amounts of information, but we really need this data to be timely. However, it still is important that we buffer these impact alerts as well, because you really want to know if somebody got into an accident, even if they happen to be in that accident at a time when, say, the 3G radio wasn't available. So it's important to employ this buffering technique at this point as well. This design is nice because it limits the overall power consumption, and the amount of 3G bandwidth consumption, and it still meets these application

requirements of being relatively timely. I mentioned that one of the sort of timeliness requirements of this application is that you want to get feedback about your drives relatively quickly after they end. But the nice thing about driving is that most people, when they arrive at their destination, their destination is a place where they have Wi-Fi. It's their home, or their work, where their cell phone is on the Wi-Fi network.

I want to briefly mention the issue of storage as a third resource constraint. These are the bytes that I mentioned in my three B's. So, these days on Internet of Things applications, it's typically not the case that you're going to be highly limited by the amount of storage that you have available. The flash storage that's available on smartphones, and even much smaller, more embedded devices, is really pretty amazing. You can have gigabytes of storage in a very compact, low power, kind of a package. So really, the main reason that you worry about storage in these applications is actually because, if you do record gigabytes of data, you have to send it somewhere. And sending gigabytes of data or processing gigabytes of data can use lots of power. So really, storage is sort of ancillary or secondary consideration, I think. But you do have to think about how much data you're collecting and what you're doing with it. So, just to give you one example of some simple things that you can do to reduce the amount of storage that you have, which will in turn improve the bandwidth and the other aspects of power consumption, in DriveWell we were able to employ some very simple lightweight compression techniques, as well as some techniques to discard portions of the data where nothing interesting was happening, where the phone was essentially idle, or not moving. And we were able to reduce the amount of data that we collected by about a factor of four. And this was relevant not because we had a storage issue on the phones and the embedded tags, but because we were able to limit the amount of data that we transmitted back by a factor of four.

So just to summarize what I told you about resources. Energy and bandwidth really are the key limiting resources in these Internet of Things applications. Hopefully you've seen that Internet of Things applications are really about engineering trade offs. For example, there's a fundamental trade off between the battery life of a device and the amount of data that you're going to transmit off of it. Because oftentimes, using the radios is going to be the primary consumer of energy on these devices. Secondly, there's often a trade off between things like latency, how long it takes to get data, and the resolution of the data that you send. If you capture the highest resolution data from these devices, especially if these devices have things like cameras or microphones, it may take you a long time to get that data off. Finally, you need to plan for the edge cases. For example, the fact that disconnections are going to happen or that batteries are going to run low, you need to make sure that your applications employ techniques like this buffering and this end-to-end principle of making sure the data is on its destination before you remove it, in order to provide reasonable functionality, even in the face of all the odd things that can happen in the Internet of Things space.

Data Processing and Storage (Sam Madden): Video 3 Part 1

In this segment, I'm going to talk to you about some of the issues that arise when dealing with the approximate nature of Internet of Things data. So we talked in the introduction about where this approximate nature of data comes from, but it's essentially because these sensors that are measuring the real world are really only measuring the world at discrete time periods instead of continuously measuring it over all time.

And second of all, the data is of a limited precision. It can only see so much in the data. For example, a camera only has a limited fidelity, a certain number of pixels. And so you can't see everything in a given camera image.

The third issue is that data can be wrong. So you remember from the introduction, I showed you this picture of GPS data where it looked like somebody drove through somebody else's front lawn. So this GP has data, when you look at this particular trace, it has some other issues, as well. For example, it turns out that in this particular log of data, there were these grayed-out periods here on this plot.

And this is showing that the GPS data when we recorded it was actually missing during certain time periods. So not only can these GPSs have noise in them, they can experience dropouts because, for example, the satellites that are up in the sky, you may not have a clear view of them because you drove past a large building or because the roof of your car attenuates a lot of the energy that you get from them.

So these noisy data also arise in these other settings. So this is the other picture that I showed you from the introduction of this acceleration data with this phone movement in it. So here's the user moved the phone while they were driving.

We've spent a lot of time doing in my research-- is thinking about how you clean up signals that look like this to make them look much more natural. So the cleanup that we did here is basically to, first of all, this red line here has this value at 10 meters per second. 10 meters per second, 9.8 meters per second, is the force of gravity pulling this accelerometer down.

So you have to remove that gravity effect from the data. And then this device is in some arbitrary orientation as the user drives around. The phone itself is in the user's pocket and it may be kind of flopping around in some kind of weird way, and that maybe is what we're seeing here with this noisy period here.

So what we have to do is transform the phone, the signal, so that the axes on this plot here at the bottom are in the direction of forward travel of the car and in the direction of lateral motion in the car so that we can see these events like these blue spikes in the longitudinal data where the device was accelerating, or these green spikes in the lateral data, where the device was breaking. So this challenge of taking this noisy signal with this movement in it and converting it into something that actually represents the motion

of the car is a very sort of prototypical thing that you might need to do in an Internet of Things application.

So there are some basic techniques for dealing with noise and missing data. These are things that many of you have probably experienced before-- things like interpolation. So for example, if there's a sequence of data points and one of them in the middle is missing, well, I can just sort of connect them together with a line. So I can use a technique like regression or curve fitting in order to fill in missing data.

We can also extrapolate-- that is, predict what is going to happen by following a trend line after we apply curve fitting or regression to it. A second very common technique that we might want to employ is some form of smoothing. For example, curve fitting will naturally eliminate spikes in data because it'll fit the best sort of straight line or best curved line to a particular sequence of data points. There are other signal processing techniques, like low-pass filters, that can remove noise in data while still preserving the overall shape of a signal.

Third kind of very common thing that you need to do with this data is alignment. So you have these signals that are coming from multiple devices, sampled at slightly different points in time, and you need to sort of time-synchronize those things. So many of you, perhaps, are familiar with the notion of editing a video and time-aligning audio with it.

It turns out that there are techniques that can automate the process of time-aligning those signals, such as the accelerometer lines that I showed you on the previous graph. These are techniques like autocorrelation or dynamic time-warping.

So there are many, many methods that can do this. And unfortunately, in the space of a few minutes that I have here, I can't possibly survey all of them. So what I want to do instead is to talk about a couple of examples of how we've been able to employ these techniques in our research here at MIT and to give you a flavor of how some of these techniques can work.

Data Processing and Storage (Sam Madden): Video 3 Part 2

So the first technique that I want to talk about is this idea that we embodied in a tool that we built called FunctionDB. And this is this idea that you could build a data processing system that runs queries on functions, on actual curves, rather than raw data itself.

So imagine I have a set of temperature readings, so this is temperature here on the y-axis and time here on the x-axis. So each of these points represents the observation of temperature at some point in time. So what curve fitting would do would fit a curve like this to these kinds of points. Now imagine I wanted to ask the question, at what point in time did the temperature cross this particular threshold?

If you just were to look at the raw readings, there isn't actually a raw reading at exactly that point in time. So the cool thing about this curve fitting technique is that you can actually solve for the point where this intersection happened. This is really just the time when this temperature curve reached that value of a particular threshold, and this is a very simple function that you can solve for.

So the cool thing about this FunctionDB tool is that first of all, it naturally smooths the data because it fits these curves to it, and it naturally allows us to ask questions about points where there weren't actually any readings because remember, these sensors are producing data at these discrete time intervals, and what we want to do is ask questions about any time interval. It may be even those intervals when we didn't have data. So this FunctionDB tool embodies this idea. Let me just show you a little bit more detail about how it works.

The idea is that you have some collection of raw data, and what we do with this raw data as we fit it a set of functions to it. Now, we have this thing that we call an algebraic query processor. You can think of this as something that really is solving these simple equations like temperature of t is equal to threshold-- solving for t in that equation. So your database system actually becomes a thing that's solving simple algebraic expressions, and then producing these kinds of results.

So there's a little bit of a challenge when you produce results in a setting like this because if the user runs a query like, tell me all the x and y points where the temperature was less than 20 degrees, it's unclear what the right answer to that is if you have the temperature as this continuous distribution of temperatures. How many points should I produce as a result of that query?

So in order to define what it means to even ask a question like this, we allow users to specify what we call a gridding, which says that you should grid the entire space of temperatures up into an 8 by 8 grid of discrete temperatures, and then return all those grid points where this predicate was true. So you can think of this as you've got this curve of temperature is less than 20, and you want to return all these red points in this 8 by 8 grid that satisfy that expression.

So now that we understand a little bit about what we expect to get as our answers to these queries, I can just give you a quick sense of how we actually do the query processing. For example, suppose you want to run this query you see at the top-- select* where temperature is less than 20, grid by x equals 8, y equals 8. What I'm going to show you is how we can solve this using a simple algebraic solver.

So the idea is that the user specifies this particular query, and what we can do is we can perform a sequence of filtering substitution approximation steps to rewrite this query into this expression, which says that $x + y - 20$ should be less than 0. This expression defines a line-- $x + y$ is less than 20-- and what we want to do is we want to return all of the grid points that are in that expression. Then this cool way of instead of having to iterate through all of those 8 by 8 grid points and figure out whether they satisfy this expression $x + y$ is less than 20, we can actually approximate the points as a sequence of these rectangles-- we call them hypercubes. And rather than having to enumerate all of the individual grid points, we can just directly return those bounding hypercubes that we know are less than the value. So the end result is that the user gets this expression that indicates all of the grid points where the value is less than 20, but we have an evaluation technique that allows us to answer this question without literally iterating through all the points in the grid.

So that was one example of a cool research problem where we employed curve fitting in order to improve both the efficiency of data processing, and we were able to answer some queries that we couldn't maybe previously answer by just looking at the raw data. The second example that I want to show you is a problem we call map matching, and this is to deal with this problem of noisy GPS data, like the picture that I showed you in the very first introductory slides.

So here is an example of the kind of problem we want to solve. We've got a bunch of observations of where a car was. These are noisy observations likely because the GPS was experiencing some form of interference. What we want to produce as a result, though, is something that looks like this black line-- an actual fitting or snapping of these raw points onto an underlying road network.

So the kind of intuition about how you want to solve this problem and about why this is hard is as follows. Consider these little red points here. So these are a sequence of observations actually right in front of my office at MIT of a car driving down the road. So we see this cluster of little red points here, and if we're not careful, we might think that this little set of red points actually represents the user being on this road here, even though by any of us looking at this, we can see that probably this vehicle followed some trajectory like the black line that's shown here. So if we were just to map these points onto the nearest road segment, you wouldn't get a very good answer because you would think that the user was on this little side street when in fact they were on this other road indicated by the black line.

All right, so the way we solve this problem-- and this is the way algorithms are going to solve this problem, as well-- is to look at the points that come before and after those

points. You don't just look myopically at one point and try and map it to the nearest road segment. You say, well, let's look at the sequence of points. So it turns out we can build an algorithm that does that, as well. We look at the past and future information in order to find the best match.

The way that we're going to solve this problem is something like as follows-- but there's some subtlety that comes up here that I'm going to spend a few minutes talking about. So the problems we have to solve are the data is sporadic, it's noisy, it arrives with varying accuracy, and as you can see from those data points, it's kind of clustered together in various ways so the standard canonical way that you would solve this problem is to use something called a hidden Markov model.

A hidden Markov model is pretty straightforward. The idea is that you say for every observation-- call an observation PI-- I can have a set of possible road segments that the driver might be on. So in my map it might be on that little transverse segment that we saw or it might be on the main road segment that we actually know the user is on.

So you call this the probability of being on each one of those little road segments the emission score. And then for each one of those little road segments, if we look at the next observation that we have, we can say, well, what is the next observation, where is it, and what are the possible transitions from that first observation that we have to the next observation? And we call that the probability of all those possible transitions, like we can look at how much time elapsed between these two observations and we can say, would it have been possible for a vehicle to travel from this first possible observation to the second possible observation in a certain amount of time? And whether or not that's possible or how unlikely that transition would be is called the transition score.

It turns out there's this very well known algorithm called the Viterbi algorithm that can take a sequence of these observations like this, where you have an emission core-- that is a probability of being in some initial state-- as well as a transition score-- that is the probability of transitioning to some future state-- and it can find the most probable set of actual states that corresponded to this. So in this case we have these noisy observations, and what the algorithm produced as an output is a sequence of the most likely road segments.

It turns out, though, that this basic model isn't a perfect fit for our particular application, and the reason is that in technical terms, our setting is so-called non-Markovian. What a Markovian assumption means is that given the previous observation, the next observation is independent of all the observations except for the one that came just before it. So you can consider this problem just thinking one step at a time.

And if you looked at our data, and in particular you think about the clustering of our data that I showed you, the GPS data, you can see why this observation doesn't really hold. This data often arrives in lots of little batches, these clusters that are all grouped together. And then you'll see a big jump and there'll be another cluster of observations. And so

what's really relevant are the sequence of clusters that you travel through, not the individual raw observations.

So our new method-- essentially what it does is it finds these clustered observations that are not independent, it groups them into these clusters, and then it runs this algorithm that finds a sequence of possible paths through all the clusters that we identify. And unlike the previous method where we consider this one step at a time, what the new algorithm does is it scores an entire path relative to all the observations we see. It produces a set of candidate paths that the user might have taken by considering these clusters, and then it computes a score for each one of those paths.

The way that it first of all computes these candidate observations is to consider all the possible segments that the user could have been on when they were in one of these clusters, and then, say, if there's another cluster over here, all of the possible segments that the user could have been on in there, and then it computes in a clever way using dynamic programming all of the possible paths between these segments. And once we have all those possible paths, then what we can do is look at each one of those paths, and we can look at the other sensor data that we have about them-- so for example, the accelerometer and gyroscope readings that we had from the device-- to see which ones of those possible paths are consistent with the observations that we saw, like consistent with the turning and stopping behavior of the device.

So to give you an example on a real-world trajectory what this would look like, these would be a set of clusters that we would compute. We would find all the paths between all of these possible clusters, and then we would score these entire paths using the observations that we have-- that is the position estimates-- as well as the accelerometer and gyroscope data that we have. And it turns out that this works really well. Here's an example of an output from our algorithm. Again, these little blue and purple dots are the position observations and the black line is the output of the algorithm.

But here's another one. Here the output is a red line, and again, you can see that the sequence of observations here is very, very noisy. It's kind of all over the place, but the algorithm is able to infer from the combination of sensor data and these sequence of noisy observations the most probable route that was taken. And this works in lots of different settings. Here is an example over tens or hundreds of miles on a freeway. Here's another example where we ran it using only cellular data, so these are very, very noisy observations. They're off the main road segment by sometimes as much as tens of kilometers, and still we're able to find the most probable path by considering the sequence of all the points that we saw.

So those are just two examples of how we can deal with missing and noisy data. Again, there's lots of different methods that you can employ, and you can see how we employed some of these methods in the techniques I talked about-- things like interpolation, extrapolation, smoothing, fitting models-- so this notion of using hidden Markov models or a Viterbi decoder for trying to understand the sequence of steps or states, like the sequence of road segments that a car went through, in a very efficient way. And then I

talked about these two specific use cases, the idea of FunctionDB for answering queries over functions of data, and this Vtrack system for allowing us to find the most probable sequence of road segments that a vehicle traveled through, given an input of a set of noisy data.

Data Processing and Storage (Sam Madden): Video 4 Part 1

In this segment, we're going to talk about how we detect outliers and anomalies in Internet of Things data. So in order to get you thinking about why we should care about anomalies, let me just point out that almost all Internet of Things applications really are about identifying unusual events. OK?

So think about some of these things. Rapid detection of equipment failure or degradation. Right?

You've got some big power plant. You want to know when it's about to shut down or go offline. So if you don't lose thousands or hundreds of thousands of dollars when the system is down.

So just to give you a few examples of unusual events that might arise in common Internet of Things applications. Consider monitoring a piece of equipment. Right?

What you want to do in this setting is understand when it starts behaving strangely. Right? You want to detect the fact that it's about to fail.

That means that you want to detect an outlier. You have this long sequence of really boring, nothing happening on the equipment. And you want to detect when something new starts happening because that's an indication that something is breaking.

Another example of this is physical tampering or security. Right? Most of the time a security camera or a door lock sensor that's detecting tampering is not going to be experiencing anything. It's going to be experiencing this very boring continuous signal.

What you want to do is detect when something new and interesting happens. OK? So, actually, finding unusual events is really often times the critical application for many Internet of Things deployments.

So let's think a little bit about what we would like from an anomaly detection system. So first of all, we'd like such a system to automatically detect outliers. It should tell us, for example, that some particular sensor is experiencing something unusual.

We'd also not only like it to tell us that's something unusual is happening, but we would like to have it tell us the properties of those things that are experiencing some unusual behavior. For example, maybe three devices that are behaving strangely are running a particular version of some operating system. And then finally, what we would like it to do is to rank and triage these outliers.

It should tell us which outliers are the ones that are most significant for some definition of significance. It may be that the most frequently failing devices are the ones that we should go pay attention to. So the way I like to think of anomaly detection is sort of a pipeline like this.

There's some anomaly detector which is out there telling us when an anomaly happens. And then there's a second stage, which is taking these anomalies and what it's trying to do is explain them. It's trying to tell us what are the common properties of these anomalies.

So that we can go out and either fix some issue that happens to be in our deployment, if what these anomalies are pointing out is some kind of a bug or a failure. Or allow us to understand exactly what it is that is going wrong, if what these anomalies are pointing out is that there's, for example, some imminent failure in some device that we're measuring.

And then finally, we have this ranking stage, which is going to tell us which are the outliers that are most important to pay attention to. Because if you have a deployment of thousands or tens of thousands of sensors, there are going to be a few of them that aren't working at any point in time. What we need to do is to focus on the ones that are really most broken or they're indicating the most severe problems.

I want to first start by talking about the anomaly detection problem. So there are many different ways that we might detect anomalies. I'll just give you a few quick examples.

Maybe the most obvious one is sort of a statistical definition of an anomaly. So for example, if I take a distribution of readings, like a distribution of temperatures over time, an anomaly might be one that is say, more than k standard deviations away from the mean. The nice thing about using statistical definitions like this, that are based on distributions, is that they're robust to slight changes in the overall behavior devices over time.

So for example, instead of defining an absolute temperature threshold that might be bad, what we can do is have a system that tracks the average temperature over time and then looks at devices that are experiencing something extreme relative to the average, as opposed to us having to hard code in some absolute threshold. We can define these kinds of statistical distributions in a multi-dimensional space, as well. So maybe this is a distribution of temperatures versus humidities. Right?

And we might want to understand the devices which are experiencing unusual values in this multi-attribute or multi-dimensional distribution. And it turns out just like standard deviation as a measure of distance in a single dimensional distribution, we can talk about multi-dimensional distance. This mahalanobis distance, as it's called, technically. That defines the distance from the center of some typical distribution in a multi-dimensional space.

Of course there's lots of other kinds of things we can use. For example, we might just define absolute thresholds. We might have other rules that we would like the data to respect. So conventional kinds of things in a database system, which maybe could generalize into an Internet of Things applications are things like, constraints that say no employee makes more money than his manager. OK?

So you can imagine how these kinds of constraints might generalize to a sensor deployment. Like, no temperature sensor reads a temperature that's higher than the temperature at the center of my reactor, or something. So now we talked a little bit about how we might detect anomalies, I want to talk a little bit about what we can do to kind of explain what's going wrong with anomalies.

So the idea in anomaly explanation is as follows. We're trying to find a description of the anomalous records. What are the properties of the records that are anomalous that are common to those anomalous records and not common in the overall population of data that we see? OK?

So it might be, for example, that the sensors on the eighth floor, they have a temperature that is too high. The observation here is that in complex data sets every record really has a very large number of attributes. So let me just show you very quickly.

I'm going to zip by. These are all the attributes in our DriveWell application that we have about a given drive. OK?

So we literally have hundreds of properties that describe the features of a given drive in this application. And any of these things might be the key explanation that tells us why something is an outlier. OK.

So I'm going to talk about two methods for explaining these outliers in data. The first one is a classification-based method. And then the second one is a method based on something called frequent item sets.

Classification, for those of you who aren't aware of it, is basically a way of giving it a data set, dividing it into multiple classes. Most frequently you would think of dividing it into two classes. So in the outlier problem, what we'd want to do is to find a set of attributes, that when we plot all the data points against that set of attributes. There is a line that cleanly separates the points that we're outliers from the points there were so-called inliers, not outliers.

So in this data here, if these blue points represent outliers and the red points represent inliers, you can see that if we draw this line through it, it does a pretty good job-- not a perfect job, but a pretty good job-- of separating inliers from outliers. Because this would tell us that this best separator, which has some function here-- temperatures greater than a times time, plus some constant here, b-- is the line above which points seem to be outliers. OK?

So this expression here tells us that there's something about sensors with a particular temperature in a particular time range that are outliers. OK? So we understand or explain the property. And this classifier that finds these kinds of separators is usually called a support vector machine. But there's lots of different classifiers in the literature.

Building these classifiers essentially involves searching over the space of all these possible attributes that we might build these classifiers over, and finding the ones that best explain or best separate this data.

Data Processing and Storage (Sam Madden): Video 4 Part 2

The second technique I want to tell you about is a technique called frequent Itemset mining. So frequent itemset mining is a good choice when you're working in non-continuous domains. So the example of temperature and time data that I showed you, you can think of those as being sort of ordered attributes that can be plotted along some multidimensional space.

But oftentimes the data that we're going to have is going to be more categorical. We're going to have features or properties of the data items that don't follow some distribution. So, for example, readings that were collected from a particular type of device like an iPhone 6 or in a particular country like Canada. These are not order domains, they're categorical domains.

So the idea in frequent itemset mining is as follows. We're going to look at the properties of the points that are outliers and compare them to the properties of points that are inliers. So if I give you a dataset like the one that's shown here. For example, imagine each line is a trip or something that we recorded in an application. Well, you can see here that the outliers compared to the inliers, well the one thing that sticks out immediately is that the outliers have Canada and none of the inliers have Canada, right? So it kind of looks like maybe there's something going wrong in Canada here.

So it's true that some of the iPhone 6's in the USA are also having a problem. But we see a lot of iPhone 6's in the USA that are in this inlier set. They seem to be behaving just fine. So this is maybe an indication that we should go look at these phones that are in Canada and see if there's a problem there.

So what frequent itemset mining does is it basically compares the frequency of these different sets of outliers to the frequency of the sets that occur in the inliers. So the first thing we can do is we can say, what are the common things that occur in the outliers? So in this frequent itemset mining world we call these things with a particular support. So what support means is these are elements that occur in one of these sets with more than some frequency, like more than two times.

So for example, iPhone 6 occurs in this data set three times. Canada occurs in this data set four times, iPhone 5 occurs three times, and the pair iPhone 5 Canada appears three times as well. But other things like the pair iPhone 6 Canada only occurs once. It doesn't have support 2 so it's not in the outlier set.

We could do the same thing for the inlier set. So we see for example that iPhone 6 occurs five times here, iPhone 5 in the USA occurs three times, iPhone 6 in the USA occurs five times, and so on. So what we can do in order to identify candidate outliers it's just to take the difference between this outlier set and this inlier set. And what we'll see is that, well, this set Canada never occurs in the inlier set, right? So that's a good indication that Canada is the problem. And similarly for this set iPhone 5 Canada.

OK. So this gave us a way where we could sort of take all the data, all the attributes of it, and rank them and find the ones that seem like they might be good candidate outliers. And the cool thing about this frequent itemset mining technique is it's a super well-understood and studied technique and there are very efficient algorithms for computing this, even over very large high dimensional datasets like the one that I showed you here.

So just to give you an example of a system that embodies some of these techniques for outlier understanding explanation I want to show you a demonstration of a tool that we built called Scorpion. So the idea of Scorpion is as follows. Suppose you have some visualization like this. In this case, this is a visualization of some census data. It has what is, arguably, an outlier. So, for example, this bar here for males seems to be very high. So we might like to understand what is it? What are the common properties? How do we do this outlier explanation on this set of records that were used to compute whatever this chart is showing us?

So we want to find the common properties of the points that contributed to this outlier group in order to understand why these outliers exist. So in order to understand this a little bit better I'm going to take a different example visualization. This orange bar represents an aggregation, or some combination of data from some underlying dataset. So, for example, imagine these are average salaries in some particular country. So these orange bar is actually contributed to by all of these orange points.

And the outlier detection problem. What we want to do in order to solve this is to find some predicate, some rule, that when we apply it to the orange points, if we remove those points that are contained in that predicate, will make this orange bar look more like normal. So for example, maybe if we remove these two points this orange bar goes back to normal. And that makes the orange bar no longer an outlier.

And then what we want to do is we want to say what are the common property of those points that I removed? So maybe it turns out this is Warren Buffett and Tim Cook. And it turns out their job is CEO. And so somehow they got put into this dataset and they super skewed this overall average income distribution.

So this is the idea in the Scorpion tool. And just to kind of map this onto the anomaly detection explanation workflow that I showed you before, in Scorpion the human tells us what the outliers are. A person points at a bar that they say looks bad. But what the system does is it uses this, sort of an interesting definition of, we call them bounding hypercubes in order to find and classify the outliers.

To explain outliers Scorpion uses a slightly different technique than either the classification approach or the frequent itemset mining approach that I described before. But it's pretty straightforward. Instead of trying to find a separator like the classification approach did before, what Scorpion tries to do is to find a bounding box. Something that encompasses all of the outliers and as few of the inliers as possible. Now notice that there are lots of different possible sets of these hypercubes that might be a good explanation for the outlier here.

So this particular one says that, well, the outliers are points whose temperature is between some value x and y here and whose time is between some value w and z here. But there's another possible explanation, which is that, well, there's actually two outlier hypercubes that fit the data. And so there are lots of different possible answers depending on how good or tight you want the fit to be. And so what Scorpion essentially does is it searches through the space of all these bounding hypercubes looking for ones that maximize the number of outlier points that are encompassed and minimize the number of inlier points that are encompassed. So this is a very complicated search algorithm that really was the heart of a research paper that we wrote here at MIT.

So what we're looking at here is some temperature data from an indoor building deployment. So the blue dots at the top are showing us the average temperature in a building, and the red dots are showing us the standard deviation of temperature in the same building. And you can see that the temperatures mostly look normal over the first few days, the ups and downs in the data, except for about five or six days in there's a sudden very large spike in the orange data points. We can actually highlight those data points and see that temperatures went crazy during that particular time period.

All right. So what I'm going to do now is I'm going to run the Scorpion tool. So what the Scorpion tool allows us to do is to select a set of points which are outliers. So I can highlight a set of points and I can see these points look bad. They look different than what I expected them to look like. And then I can highlight another set of points and I can see these points look good. These look like what I expected them to look like.

Then when I hit Run Scorpion what the system does is it does this analysis that I showed you on the previous slide where it finds the bounding boxes that encompass the outliers and don't encompass any of the inliers that were selected. And it takes a few seconds to produce these results. But what the system will spit out is a set of possible predicates. So you can see here that it gave us some candidate results. The first result that it said here is sensor ID is equal to 18.

So I can take that result, that reading, and I can click on it. And then I can say, now filter out all the data that is in that outlier group in the group that corresponds to sensor ID 18. And when I do that what you can see is that the readings all went back to normal. So by running the Scorpion tool what we were able to do is explain that the most common outlier property of these bad sensor readings was that they belong to one particular device, to sensor ID 18.

So in this particular example I happen to know what happened to sensor ID 18 is that the voltage on the device dropped below some threshold. And it turns out that the microprocessor here was able to actually run at that lower voltage but the temperature sensor itself went haywire and started producing crazy readings like temperature is equal to 120 degrees Celsius. So when we remove sensor ID 18's data from this distribution the data looks much more normal. So we were able to explain that sensor ID 18 was the bad thing that was going on here.

To conclude, Internet of Things data is quite different from the kinds of data that you encounter in other data processing scenarios like web applications. The fundamental reason for this is because, first of all, the data is approximate. It's only a sampling of the real world. And second of all, because of these issues related to noise. In addition, what I've told you about is that many Internet of Things applications have special requirements. For example, you have to deal with these very limited resources like battery or bandwidth. And you need to deal with a kind of very particular sort of data processing, which involves finding the unusual outliers in the data and then explaining where those outliers come from.

I hope you've enjoyed this segment on data processing and Internet of Things. Thank you very much.

Localization (Daniela Rus): Video 1

Hi. My name is Daniela Rus. I'm a roboticist, and the Director of the Computer Science and Artificial Intelligence Laboratory at MIT, or CSAIL. I'm also a professor of electrical engineering and computer science at MIT. And today, I'm very happy to tell you a few things about localization in the IoT world.

Now the Internet of Things has the potential to awaken 99% of the devices around us, of the inanimate objects around us. And in the process of doing this, it will connect these devices. It will provide them with sensing capabilities, with communication capabilities, with computation capabilities. So the whole world order will change as a result of all this. It is predicted that by 2020, we will have over 50 billion devices all connected to each other and to us. That is a lot of devices.

Now if you do the math, each one of us will individually have at least seven devices connected to the internet. And actually, if you think about what is happening today, some of us already have more than seven connected devices in their homes. For instance, in my home, I have 16 connected devices. And I can do very interesting things with these devices. For example, I can find out where every member of my family is by using tools that are already available on my smartphone app.

But in the future, we can do so much more. For example, shopping might look like this. Say I want to experiment with a new recipe. I go to the store, and my phone, connected to my fridge and my pantry, could figure out exactly which ingredients I am missing so that I can deliver successfully on my great new recipe. Perhaps I'm out of apple sauce and peanut butter. Well, my phone would direct me precisely to the location on the shelf where I could pick up the peanut butter, and the location on the shelf where I could pick up the apple sauce. And because peanut butter jar and the apple sauce jar could talk directly to my phone to inform my phone about where they're located.

And outdoors, our cars will be the most intelligent nodes in the Internet of Things. They will be connected to each other. They will be connected to people. They will be connected to the cloud. They will be connected to the internet.

Now all of these devices will have to instantaneously localize themselves. They will have to have a sense of identity. And they will have to have a sense of the surrounding world. And this is where localization comes in. So in this module, we are going to look at some basic techniques for localizing Internet of Things devices that have the capability to sense, compute, and communicate. We will look at how to make these algorithms robust, and then we will look at a case study outdoors, and a case study indoors. So let's get started.

Localization (Daniela Rus): Video 2 Part 1

The localization problem can be formulated as follows. Given a device that leads to localize and given a reference frame, this device must, through sensing, compute its position and its heading in the world. The [INAUDIBLE] represented by position and heading is referred to as the pose of the object.

So, in this slide, you can see an object whose location is marked through a point in the xy-plane. And its heading is marked as an arrow. So the localization problem is to determine the distance from the origin of the system of reference to the point and the vector that shows the heading of the object.

One option to do this is to do something called open loop pose estimation. This means that there is no sensing involved in the computation, and one solution that we often employ with wheel-based devices is to use odometry. Counting the number of rotations of your wheel, you can approximately estimate where the device is.

So, in this picture, we see a map of the infinite corridor at MIT. And we see a robot tasked with traversing this map for a long period of time, for several hours. Right in the middle, we see a drawing of the position of this robot, as computed through odometry. So you can see that it looks nothing like what the map looks like.

So why is this the case? Well, this is the case because open loop localization is fraught with error. There is error because the wheels slip. There is computation error. And this error accumulates over time. So open loop pose estimation is simple and straightforward, but not good enough.

So what can we do? The most common way to estimate the location of an object in 2D is to use some form of sensing. We can sense distances or we can sense angles. So we have range-based localization and bearing-based localization.

Localization (Daniela Rus): Video 2 Part 2

So how can we do better for localization? Well we can introduce sensing. And the most common things to sense are distances and angles. In other words, we can have range-based localization or bearing-based localization.

For sensors, there are a multitude of devices we can use. Range can be measured with things like sonars and IR sensors. Bearing can be measured with a compass or with computer vision. Range to point can be measured by received signal strength in Wi-Fi enabled devices, or by time of flight for acoustic, or radio beacons. Or they can also be measured as the difference in time of arrival between two signals-- for example, an acoustic signal and a radio signal.

And range and body-relative bearing to an object can be measured using a radar, using a laser scanner, or using a computer vision stereo set. All these sensors are available off the shelf and can be used to help us improve the localization for our IoT devices.

The classical approach to measuring location is triangulation. And this is an old technique that has been around since antiquity. In this set up for triangulation, the device measures either the range to a landmark, or the bearing relative to a landmark. Given the device's heading, we can measure angle theta, or we can measure the distance to landmark L.

So here's how the algorithm for triangulation from range data works. The device is assumed to be at a location p, which is unknown. And the device can measure the distance to two landmarks, L1 and L2. And say these distances are d1 and d2. So given d1 and d2, and the location of L1 and L2, can we compute the coordinates of p?

Well, since L1 is fixed, and L2 is fixed, we know that the device is located on a circular arc of center L1, and also on a circular arc of center L2. Therefore, p is at the intersection between L1 and L2. And we can compute this in closed form.

The easiest way to see the computation is to consider shifting the system of coordinates so that the origin is at L1 and the x-axis goes through L1 and L2. In other words, it's defined by the segment L1 and L2. Given that L1 is at location 0, 0, and L2 is at location a and 0, we can use straightforward trigonometry to compute the x-coordinate and the y-coordinate for point p in this local system of coordinates. And then we can use the transformation between the global system of coordinates and this local system of coordinates to get the coordinates of p in the global reference frame, which is what we want.

So, a natural question is, are we done? What do you think? Are we done? Well, actually, we're not quite done. Because we have computed the position of the device, but not the heading of the device. If we are talking to our fridge, and we don't really expect the fridge to move, then we are done. Because all we are interested in is the actual position of the fridge. In other words, the x and y locations computed with the trigonometric formulas we have seen.

However, if the device is a mobile device, perhaps a self-driving car driving through a road network, or a hospital delivery system that is taking medication to various patients, then knowing the heading of this device is critically important in how we reason and predict about what the device will do next.

So are we done with the location computation? Well, the answer is no. And the reason the answer is no is because in general, when we compute location by intersecting two circles, we end up with two options. We end up with point p or point p prime. So how do we know which one it is? It's important to disambiguate between p and p prime.

And we can use additional information. We can use a map. And we can keep track of how the device is moving in the world, on the map. Another option is to keep track of the history-- to keep track of where the device was a few minutes ago. And that will help us disambiguate between the two points.

And finally, a third option is to add another point in the computation, to measure the distance to a third landmark, L3, and to do pairwise intersections between all the circles that are now involved with the centers at L1, L2, and L3. And the intersection between three circles will be a single point, provided we have accurate measurements and accurate computations.

So are we done? Well, if the question is to find the location of the remote control, then we are done? Because the remote control is not expected to be moving continuously. And knowing the coordinates of the remote control is sufficient to understand everything we need to know about the remote control.

However, if the problem is to find the location of a device that moves, perhaps a self-driving car, or even a manually driven car with a person inside, connected to the internet, or a delivery robot that takes medicine to various rooms in a hospital, then we are not done. Because knowing the heading of the device is very important for reasoning and predicting what the device will do next.

In other words, we also need a method to compute the heading if mobility is involved. So, here's our robot or device at location p. And it can be in any of these orientations. And it actually matters which orientation it is, if the object is mobile.

Now, how can we infer this location? We can use motion to compute the difference between two positions across time. We can also use extent-- using two ranges measured over the device baseline.

Localization (Daniela Rus): Video 2 Part 3

We can do the same type of triangulation using a different kind of sensory measurement. In other words, using angles, using bearing data.

In this scenario, the robot observes two landmarks, L1 L2, and computes the bearing relative to landmark L1, and the bearing relative to landmark L2. So a natural question is, are two bearings enough for localization?

Well, let's see how this could work. Well, given two fixed points, and a fixed angle between those two points, the robot could be here, or here, or here, or here. The robot can be anywhere on the circular arc defined by the angle and the two fixed points, L1 and L2.

So let's stop for a minute and ask ourselves, can the device be anywhere on the circle, or just on the circular arc?

Well, the answer is only on the circular arc. Because the red circular arc contains the points with relative bearing alpha prime, which is equal to pi minus alpha. So we have just the circular arc.

But there are lots of points. There's an infinite number of points along that arc. Where can the robot be? Well, just like in the previous case, we can resolve this question by adding another landmark in the system. So let's consider a setup where the robot can measure landmarks L1, L2, and L3. And let's consider the circle defined by L1 and L2. And let's consider, also, the circle defined by L2 and L3. The device is at the intersection between those two circles. Now, the device is necessarily at one of the intersection points, because the other intersection point is L2. So that's all we need.

In other words, we do not need to do a third circular intersection in order to find the x-y location for the robot. And all of these circles can be computed using very straightforward trigonometry.

Now, just as in the case of computing location from ranges, in the case of computing locations from bearings, we can also determine the heading of the device. And this is done by considering pairwise intersections between all the circles that we now have available.

So we use the circle defined by L1 and L2, and the circle defined by L2 and L3, to compute the x-y locations. We can now use the circle defined by L1 and L3 to compute the heading of the device. And with this we have a complete solution for the localization problem, when we can measure three distances, or three angles, to three landmarks, whose locations are known in the environment.

However, one important assumption for all these methods is that the measurements are precise. If the measurements are not precise, then we are in a little bit of trouble, because all those circles will not intersect perfectly.

Triangulation from bearing, and triangulation from ranges, works beautifully if all the measurements are precise. However, for real world cases, where objects use real sensors in order to measure distances and angles, we will have noise and uncertainty. And in this case, because the sensors are imprecise, we will have to deal with inaccuracies. We will have to deal with inaccuracies in distance, measurements, and in angle measurements.

If we are working with a physical sensor for distances, the maximum error associated with that sensor is usually specified as part of the technical specifications of the sensor. So knowing that number, we can then refine all of our calculations about the location for the object as follows.

We know that the object will be located in the region defined by two circular arcs. One circular arc, assuming accuracy for the performance of the sensor, and the other circular arc, assuming maximum error for the sensor. So we can carry this observation on how we define the circular arcs for each of the landmarks, and then, instead of computing a single point that defines the location of the object, we can compute a region where the object is very likely going to be. So when we have imprecision, instead of working with points, we now have regions that approximate the location of the object.

In the case of bearing, the measurements will have angular noise, or angular error. And again, we will have a maximum range for the error. So we can use the same idea to define two circular arcs. One arc, where we assume that the device computes the location accurately, and one arc, where we assume that the device computes the angle with maximum error. And again, we can define a region that is subtended by two circular arcs. And this region can be computed in closed form, using basic trigonometry.

You can see the region right here. We can do the same computations, relative to every pair of landmarks. And by intersecting these regions, we compute a new region where the object is likely to be located. So again, instead of estimating the location of the object using points, we now estimate the location of the object approximately using a region.

In this segment, we have defined the localization problem. And we have looked at how to use triangulation in order to compute the location of an object. Now, we have defined the pose of the object as the location and heading of the object. And we have shown an algorithm for computing the pose using distances, and another algorithm for computing the poses using bearings.

We have also looked at how uncertainty plays into these computations. And it's important to consider uncertainty, because IoT devices live in the physical world, and, therefore, their measurements will have intrinsic error and uncertainty.

Localization (Daniela Rus): Video 3

In the previous segment, we have looked at how to compute the location of an object using triangulation. And we saw that by computing the distance or the range to three landmarks whose locations are known, we can determine the complete [? pose ?] of the object. We can determine its location and its heading. But we have also observed that many of the devices we have to deal with are likely to give us erroneous measurements, so dealing with uncertainty is important.

In this segment, we will look at an algorithm that is guaranteed to compute a robust localization solution. In other words, it's guaranteed to not make a big mistake in computing the coordinates of the object. And this algorithm does not need the assumption from the previous algorithms, so the previous algorithms assumed three fixed landmarks of known location.

This algorithm is also range-based, and the algorithm is guaranteed probabilistically to be robust with respect to noise. The algorithm does not need beacons. In other words, the algorithm does not need the L1 L2, and L3 we saw in the previous segment. And the algorithm works even if the nodes are moving.

So we assume that we have some nodes that can talk to each other in the network. These are the green and the red nodes in the picture. And we assume that the nodes can talk with each other and can measure the relative distance to each other.

Now given these measured ranges, we would like to establish a system of coordinates and we would like to establish the coordinates for each of the nodes within the system of coordinates. So let's say the nodes talk back and forth and measure relative distances to each other. We would like an algorithm that, bingo, figures out all these coordinates as you can see them in the picture. You can get more details about this algorithm by looking at our census 2004 paper.

Robustness refers to our ability to guarantee that the locations are accurate probabilistically. And what does that mean? What are the issues that may arise? Well, remember we measure the location of an object by intersecting two circles. So, and the critical point in that computation is to pick the correct intersection point.

Now imagine we have a small error in the measurement of one distance that leads our algorithm to pick the wrong intersection point. This small measurement error could translate into a huge topological error in computing the locations for our system of nodes. And that is the main problem that we need to worry about, that we need to defend against. And this kind of error is called a flip ambiguity. So the objective here is to come up with an algorithm that is guaranteed to not have flip ambiguities.

The theoretical basis for this algorithm is something we call a robust quadrilateral. So consider the graph in this picture. The theoretical basis for robust localization is the robust quadrilateral.

And there is a topological basis for robustness that essentially includes three properties. We need rigidity. We need no discontinuous flex ambiguities. And we need to probabilistically constrain the quadrilateral so as to minimize the likelihood of a flip ambiguity. These robustness characteristics come out of the topology literature.

Here is the basis for doing trilateration using robust quadrilaterals. The idea is as follows. If three nodes of a quad have known position, the fourth can be computed by trilateration and quads can be chained in this manner. Once you have identified the location of three nodes, you can use the subset of those nodes in order to compute the location of our additional nodes.

And the exciting thing about our paper and our algorithm is that we can actually put a bound-- we can actually come up with a measurement that tells us whether a quadrilateral is robust or not. And this measure involves the relationship between the minimum edge length and the minimum angle of each of the four triangles that are contained inside the quadrilateral. And you can see that relationship right here.

So how does this algorithm work then? Well, we take our points, we define the quadrilaterals. In other words, given the relationship between the minimum edge length and the minimum angle, we can take a quadrilateral, we can check the minimum edge lengths and the minimum angles in the quadrilateral and we can determine whether that quadrilateral is robust or not. So now we have a test, in fact a very simple test for checking whether four points are useful for localization, for robust localization.

So here's how the algorithm works. We start with one point and we call it the origin of the system of coordinates. We pick another point and we define the x-axis. We pick another point and we define another axis. We pick a third point and we define the second axis. And now we have the coordinates of three points in this new system of coordinates that we have defined.

Given this system of coordinates, we can incrementally add points. We determine whether the quads defined by the four points are robust or not. We scrap the points if the quads are not robust. And we keep the points and we apply trilateration if the quads are robust. We can do this by chaining quads. And altogether we have our solution and we have a guarantee that in the solution there will be no topological mistakes in how the coordinates are computed.

So the great news is that we now have a very simple algorithm that is guaranteed to compute robustly locations, even in the presence of mobility, even when we do not have landmarks. We have implemented this algorithm on a variety of platforms. And the following video will show you a particular implementation which is done using robots and MIT Cricket nodes.

DAVID MOORE: Hi, I'm David Moore, a graduate student here at CSAIL. In front of me I have five Crickets, which are wireless sensor nodes developed by the networking and mobile systems group here at MIT. They can act sort of like an indoor GPA system in

that once the devices have been placed around they'll self-configure their own position and discover the 3D position of each node using the technology.

I'm going to show a demonstration where I track a mobile node, in this case a Roomba autonomous floor vacuum using the real-time positions discovered by this sensor network.

PROFESSOR: I hope you enjoyed the video. We collected some data to see how well the Roomba robot is able to localize using robust quads. And here you see data from one of our experiments.

On the left, you see the trajectory of the robot. In blue, you see the localized path. And in red, you see the true path, which is computed with an external localization system. On the right, you see the error in the robot path over time.

So you see that the error remains low over time. And this is really exciting, because in many cases error accumulates as time goes by. We have also done a number of experiments in simulation using lots and lots of nodes.

And here we see a comparison between using localization with quads and doing localization without robust quads. On the right, you see the results of localization with quads. On the left, you see localization without robust quads. The little lines you see attached to points show the error that we have observed for localizing each of those nodes.

So what can we see here? Well, we see that the error for the case of localization with robust quads is much smaller than the errors that we see in the case of localizing without robust quads. But we also see that some nodes are not localized using robust quads, whereas all the nodes are localized with a regular method.

So what is going on here? Well, the red nodes, the unlocalized nodes ended up not being part of any robust quad. Because of that they couldn't be localized. So the trade-off with the algorithm is if you have a set-up where the nodes define robust quads, you get really good location numbers. But if you don't then you simply can't localize the nodes.

Are there any alternatives for these nodes? Well, the answer is yes. We can do something additional. So for the nodes that are not localized, perhaps we can bring the big guns. Perhaps we can bring a big node that has differential GPS or it has some other powerful localization method, and we can use this node to transmit coordinates and localize those individual nodes. In other words, we can use an external localization source to take care of the handful of nodes that remain unlocalized using the robust quad method.

So in this segment we have looked at an algorithm for computing robust location for nodes without beacons, without known landmark locations using distance measurements and supporting mobility. The method is very simple and it uses a very lightweight computation that amounts to examining the angles and the edges in a quadrilateral.

Localization (Daniela Rus): Video 4 Part 1

So far we have looked at basic methods for computing location. We have looked at methods that compute location relative to landmarks and we have looked at the methods for computing location independent of the presence of landmarks. Now we are going to look at a few case studies. So in the first segment, we will look at a case study with outdoors localization.

So question here is, what if we are in a place where we have no landmarks but we have a map and we are outside? Well, a simple answer is we have GPS, so why not use GPS? We all know that GPS gives us location information. Well, GPS gives us so-so location information. In other words, GPS might introduce too much error.

And GPS is particularly notoriously bad in urban canyons. So in city environments, we cannot really rely on GPS to figure out where we are. And I'm sure you have experienced that yourselves. So if we're outside, we have GPS, but it doesn't work well. If we're inside, we don't even have GPS.

So what to do? For this case study, I would like to focus on our autonomous vehicle which is an electric [INAUDIBLE] car developed by my team in Singapore. This picture shows one of our self-driving cars. This car has been developed by our team at SMART in Singapore. And what you can see in this picture is the car and its sensor suite.

The car has a tilted down LIDAR which is used for localization. It has a planar LIDAR, which is used for object recognition. The car has a webcam which is used for a variety of purposes because it provides the visual feedback. And the car also has an IMU sensor which is used for odometry. Now this car is capable to drive autonomously on roads, but the car needs a map.

And here's how the control system of the car works. There is a loop that perceives the world through sensors. Figures out the location of the car on the road. Plans the next step of the car. And then executes it.

A critical point in this loop is to understand where the car is located. We divide vehicle localization into stages. In the first stage, we develop algorithms using the curb and using the structure of intersections. We look for curved features and we localize the vehicle using the map and the features detected by LIDAR sensors. In the second stage, we extend our methods to consider the surrounding world, the full three-dimensional surrounding world, and we look at landmarks that happen in the vertical plane as well as the planar landmarks we get from curbs.

So curb features are very valuable in figuring out where the car is, but they're not sufficient. We also use other vertical obstacles as features. So here's why a curb is very interesting for localization. You see the curb has a lot of interesting features, and when you approach an intersection, the topology changes, all of these things can be detected with a LIDAR sensor and compared against the previously stored map to figure out

where along the roadway the vehicle actually is. The localization stage is based on the Monte Carlo methods.

We call it the MCL method. And localization has three stages. Particle propagation, importance weight updating, and resampling. The first two stages use the map, the odometry of the vehicle, and data from the LIDAR. Together these stages compute an estimate.

The best estimate for where the vehicle is located. The tilted down LIDAR on the vehicle is used for curb intersection feature extraction. Now the results are then filtered through a second order differential filter, and then the boundary points can be extracted, as you can see in this image. These boundary points are local minima that come out of the filter. The extracted features can be used to build two different types of virtual sensors.

There is a virtual LIDAR sensor for curb features. The sensor accumulates and fuses points along the curb. And then there is the virtual LIDAR sensor for road intersection features, and this is triggered when there are no longer curb points. Together these two sensors can compute the location of the vehicle along the road network. We have implemented the Monte Carlo localization method on our self-driving vehicle, and here are some results.

In cyan, you can see the true curb outline. In green, you can see the location of the vehicle computed by GPS. In yellow, you can see the location of the vehicle computed by odometry. And in red, you can see the location of the vehicle computed using our method. So we know that odometry is very bad, just like we said in the beginning of this module.

But we also know that GPS is pretty bad. And this is not even a dense, urban canyon. This is a campus with some buildings, but not very tall buildings. However, note how close the red curb is from the cyan curb. That means that the Monte Carlo localization method using curb and intersection feature detection works very well.

We have measured quantitatively the error that we obtained on different segments of the path of the vehicle. And you see those segments. In the picture you'll see s, a, b, c, d, e, f, and g. And in the table, you'll see the average error aggregated over several runs for each of these rolled segments. We see that the error looks very good.

In other words, the error is small except in two cases. Segment e and segment b. So what is going on? Well, let's look a little bit more carefully at the topology of this path. We note that segment e and segment b are very long, straight segments.

That means these segments have not many local features. There are not many wiggles around the path of the robot. And for that reason, there isn't enough information to accurately position the vehicle. In this overlay, you'll see in red the estimated location of the vehicle for segment b. And you see that the red cloud is fairly long.

That indicates the fact that we have a fair bit of uncertainty in the location of the vehicle as computed by the Monte Carlo localization method. And this is because we do not have enough features to localize more accurately. Let's consider segment c. Here segment c contains an intersection, and we see that the spread of the red point cloud is much smaller. That means that our confidence in the localization of the vehicle is much stronger, and the region of uncertainty, much smaller.

And finally, for segment d, we see some in between. We see in between the straight, long segment and the segment with a lot of features. So we have implemented this algorithm for localization on our self-driving car. And here you can see the trajectory of the car.

You can see the laser scan that computes the boundaries of the road, and you can see in blue the edges of the road. And you see in red, the vehicle belief for where it is located. As the vehicle travels the road network, you can see the belief grow or shrink according to how feature rich the area of the road is. The more features, the better the localization.

Localization (Daniela Rus): Video 4 Part 2

So clearly our implementation shows that we have an easy-to-implement algorithm that is rich in computing reasonable beliefs for the vehicle localization. But the algorithm has some limitations. The algorithm assumes that curb-intersection features exist. And the algorithm has significant uncertainty for road segments that do not have many features. In other words, for straight road segments, the performance of the algorithm is less accurate.

Well, what can we do? The idea is to bring more information in the system. In other words, to use the vertical surfaces to compute better localization positions. And here, the intuition is that all the trees and buildings and fences that we see on the side of the roads, all the things that are likely to be captured on a map, are a significant source of additional information that will enable the vehicle to localize better.

So our extended solution is something we call synthetic radar. And the solution has two components. There is a 3D perception module and there is a 2D localization module. These two modules are bridged by a 2D synthetic radar. To 2D localization module also uses a map.

So what is going on? Well, intuitively, what we would like to do is to extract vertical features and project those features into the plane computed by the 2D localization method. We do this using a rolling window. This is a virtual window that is moving together with a vehicle.

New input is added and all input is dropped over time. But we know it's fixed in size. And on the right here, you see an example of what this moving window might look like. And you see some of the vertical features that can be extracted using a very simple laser scanner.

Accumulated 3D data is seen in this slide. And it's color coded according to intensity. Now, once we compute all these points in the moving window, we need to classify them. We need to identify whether they belong to some features or not. Any classification method it's fine.

We happen to use surface normals because we find this to be a very simple, straightforward method. And since we're applying our solution to urban environments where we expect to have vertical buildings and trees, we find this solution effective. But you can use other methods.

Given a collection of 3D points gathered through the moving window, we can identify features. And then we can identify the surface normals of these features in order to classify the points. We choose surface normals as a means of classifying the points. But any other method will do.

And so here in this picture you see our extracted points. And there's surface normals used for classification. And you see what they correspond to in the physical setting.

Given these calculated points, we can further project them onto the plane defined by the 2D localization using curb features. And now we have a planar characterization of the location of the object that includes curb features extracted with a 2D localization system. And it also includes projected vertical features extracted using the 3D perception module. And with this, we can perform much better on localization.

So in this movie, you see the physical world. You see the world viewed through the camera installed in the robot on the left. And you view the world according to the localization and features computed by our algorithm on the right. It turns out that this method greatly improves the results of localization.

Here is some data from our computation. We see that, in general, localization using this approach has consistent error. And it has much smaller error than the method that uses only 2D the points extracted from curbs.

So we have implemented this method that combines Monte Carlo localization using curb and intersection features with vertical features and that we call synthetic LIDAR based localization on the same road segment we saw before. And we observe that the error is much more consistent and much smaller than using just a Monte Carlo localization method. Specifically, remember those road segments defined by points S, A, B, C, D, E, F, and G? The error on the B segment was especially large because the B segment corresponded to a straight path of the roads that did not have too many [? curb-based ?] features. By bringing vertical features into the computation, the error gets significantly diminished because now the car can use vertical features in order to localize itself with finer granularity.

So we are very excited because this is a method that uses a simple computation and yet delivers accurate results in real time. So to wrap up, we have presented a case study for localization self-driving cars. We have looked at methods that use curb features, intersection features, and vertical features combined in the form of a synthetic radar algorithm. And we have shown that this method can compute accurately the localization information needed by a self-driving car in real time.

In this video, you'll see our final round of the algorithm. And you see that the vehicle is capable of localizing itself with great precision.

Localization (Daniela Rus): Video 5

So we have seen a case study for localizing IoT devices such as self-driving cars outdoors. And now, a natural question is, can we do it indoors? Well, the previous methods require the map and its required features, its required curbs, and its required vertical features. And it's much more difficult to acquire this information indoors than it is to acquire it outdoors.

So what can we do for the indoors case? Everyone would love to have a GPS indoors. And in fact, there is a lot of activity on the business side, and there is a lot of activity on the research side. And yet, there is no ideal solution.

All the solutions, whether they are commercial or research-based, require either too much infrastructure, or too much setup time, or specialized hardware, or way too much computation. You can hear about indoor localization using a special technique called SLAM from Professor John Leonard. But in this segment, we will talk about a new way of computing location information using intrinsic Wi-Fi signals that just exist in the world.

So here's the problem set up. We would like a user to arrive at a new indoor location and using an off-the-shelf device, such as a phone or an iPad, we would like this user to just know where they're located, perhaps, at this position in a map that describes the space. So how can we achieve this? Well, in fact, it turns out we can achieve this with great accuracy.

Here's our user entering the space. And with her device, the user is able to figure out where exactly she's in the space with an accuracy of about 28 centimeters, which is pretty extraordinary. Now, she did something really special, if you noticed. She rotated her device.

So what is going on? Well, this is the basis for how our method works. Recall again triangulation from bearing. If the user walked into our space using a special device with off-the-shelf Wi-Fi but our specialized software, and if this device knew the location of two base stations, the device could measure the bearing to those two base stations. And if the device had many antennas, the device could use each antenna to measure a different bearing from a different position. And then it could integrate everything in a computation that accurately positions the device.

However, this requires many different antennas. And such a device does not exist today. Most devices have maybe two antennas inside but not a lot of antennas, which are needed in order to accurately disambiguate using triangulation and bearings.

So the idea here is to have the mobile device emulate the antenna array. Imagine the following scenario. If you had a multi-antenna array, you would hear position information at different points along the surface of the device. The device would be stationary.

Well, since we can't do that, what if we rotate the device in order to emulate the antenna array? And that is exactly the intuition behind our indoor location algorithm. By moving the device, we can emulate a multi-antenna array.

In order to use this method, we need to know the location of several base stations distributed throughout the environment. We need to have a tablet or a phone equipped with off-the-shelf Wi-Fi cards. We need the special software that is able to emulate a multi-antenna array. And we need the user to be cooperative and agree to rotate the device in order to do the multi-antenna array emulation.

So what can we do with it? Well, one thing we can do is to localize the user, and, in fact, to localize the user very precisely to on the order of tens of centimeters precision. But we can do much more.

We can also use this method to geotag objects. So here's our user entering a library. And in this library, the user finds a book of interest.

Using our method, the user can take a picture and use the combination of computer vision techniques and the multi-antenna array emulation localization algorithm in order to compute the physical location of the book and to record it as a geotag. The computed location can be shared with friends. Or it can be used by our user, when she returns to the library, and she wants to find the same book.

So what else can you do? Well, you can do object geotagging. For example, if you walk through a space, and you notice that something is broken, say a lamp is broken, all you have to do is point your phone that is running the special algorithm at the object. And Facilities is immediately informed of the precise location of the broken object. And you can use your imagination to see how these methods apply to your application domain.

Localization (Daniela Rus): Video 6

So in this module, we have looked at various aspects of the localization problem. We have defined the localization problem as computing the pose of an object, and this includes identifying the location of the object, and the heading of the object, which is important for objects that move. And it's not important for static objects.

We have looked at triangulation algorithms that use bearing or ranges. We have seen the basic algorithms. We have seen what happens in the presence of noise. We have seen that these algorithms require at least three known landmarks in a space.

Next, we have looked at an algorithm that can compute the location of a moving object without the need to know landmark locations. Furthermore, this algorithm is robust to sensing noise.

Next, we looked at two case studies. One outdoors and one indoors. For the outdoors case, we looked at localizing self-driving cars, using a prior map, and using features extracted by monitoring the curb, road intersections, and also features computed from the vertical objects that are encountered along the road.

We have seen that we do quite well with just the curb and the intersection. But by introducing the vertical features, we are able to localize accurately, even along straight road segments where the curb does not contain a lot of information.

Finally, we have looked at a case study for indoor localization, and we have seen that by emulating a multi-antenna array, we are able to compute accurately indoor locations using known locations for a handful of base stations, using special software that can emulate the antenna array, and asking the user to rotate the device in order to emulate an antenna array.

We have seen several applications of our indoor localization method to tracking users through indoor spaces, and through object geotagging. In conclusion, there is a new world order that is around the corner and it's due to the explosion of IoT devices. It's due to the explosion of devices that are capable to sense, compute, and communicate. But these devices need to be able to localize themselves instantaneously. They need to be able to have a sense of identity, and they need to be able to have a sense of their surrounding world.

Localization is the fundamental technical component that enables all these capabilities. As the IoT field evolves, we will have increasingly more devices connected to each other, connected to us, and connected to the internet. That means that all the machines surrounding us would get smarter and they will get much more engaged with us. All these devices will need to know where they are in order to help us know more about ourselves and our world. Thank you very much.

Security in IoT (Srini Devadas): Video 1

Hi, I'm Srini Devadas. I'm a professor of computer science at MIT, and I'm going to give you a lecture on security issues in the Internet of Things. I want to talk to you first about why security is a hard problem, and especially so in Internet of Things systems. I want to talk to you about the diversity of threat models that an IoT system designer faces with respect to the power of an adversary, and the kinds of things that the adversary could do to break the security of devices or systems.

And then I'm going to describe to you defensive strategies, mechanisms that system designers can use in order to produce Internet of Things systems with guarantees of privacy, integrity, and security. In particular, I'm going to talk about three different defensive mechanisms, one of prevention, one of resilience under attack, and finally, at third corresponding to detection and recovery. And then I'll summarize and leave you with my thoughts on the status of the design of IoT systems.

I want to talk about particular attacks that are not necessarily restricted to IoT systems, but certainly can be viewed as attacks on individuals or users of IoT systems.

Ransomware is an example of an attack on an individual where an individual is threatened with the loss of his or her files because a virus or worm attacked the computer system associated with individual, be it a tablet or a laptop, and managed to get hold of permissions that allowed the virus to encrypt all of the files associated with the person, and delete the unencrypted files, and set things up in such a way that the user has to pay \$500 in Bitcoin or some other digital currency to the n ominous attacker in order to get the secret key to decrypt his of our files back. This can happen to anybody simply by clicking on a website that the user should not have clicked on.

The second kind of attack is an attack on services. A famous example of this attack is the attack on Target, which resulted in the CEO resigning in 2013. Servers are particularly attractive to attack when it comes to malicious adversaries who want to inflict a lot of damage or want to make a lot of money, because they store a whole host of information associated with tens of millions of customers.

In fact, the target server stored 40 million credit card numbers, along with personal information associated with each of the credit card holders. And the attackers managed to get hold of this information, inflicting over \$200 million worth of damage on Target and its customers. The sad fact is that there is available hardware technology, namely chip-on-card, that would have protected all of the target customers from identity theft, even though the malicious attacker got hold of their credit card numbers. Target's now learned its lesson, and now mandates that all of its customers use this chip-on-card technology in order to protect themselves.

Perhaps the scariest attack on cyber infrastructure, and an attack that is very relevant to IoT systems, is the Stuxnet attack from a few years ago. Stuxnet was a worm that attacked 14 industrial sites in Iran, including a nuclear facility. In particular, the Stuxnet attack inflicted physical damage by causing nuclear centrifuges to spin out of control,

resulting in instability in the physical infrastructure. It could have caused a lot more damage had not Kaspersky Labs tracked it down a few months after it started attacking Iran's facilities.

I want to give you a little more detail on the Stuxnet attack to give you a sense of how IoT systems could potentially be attacked through a combination of insiders and heft of private information and knowledge of computing system infrastructure. The way the virus entered the system was through a USB stick that was brought in by a malicious insider. Next, the virus propagated through the computer system and tried to discover exactly what processor it was running on.

Once I discovered the particular type of processor, it went back to its host, the writer of the virus, and downloaded information that was put into the particular processor system so it could control the infrastructure that the processor controlled. And then the virus morphed itself into a control algorithm that would cause the centrifuges to accelerate and decelerate beyond specification. This resulted in instability of the physical infrastructure and, finally, caused the entire facility to shut down.

What makes these attacks successful? It's because security is a hard problem. It's hard to guarantee security, and the main reason for that is that security is a negative goal. It's easy to say that someone can do something and verify that that person can do something. For example, I can say that Frans can access grades.txt. And I can ask Frans to access grades.txt, and if he can do it, I'm done. I've shown him and everyone else that Frans can read the file.

But suppose I want to say that Nick cannot access grades.txt. This is what we call a negative goal. I now have to make arguments associated with the restrictions that are placed on Nick that disallow him from accessing grades.txt. But the fact is that there's many ways that Nick could get around these restrictions, and I have to make the argument that Nick cannot get around these restrictions, whatever he does.

Let me some examples on how Nick could get around these restrictions on him accessing grades.txt. For example, he could change the permissions on grades.txt so it's readable by everybody, including him. Or he could walk away with disk that the file is stored on and access the bits of the file at his leisure. Another way that Nick could access grades.txt would be to attack web.mit.edu and get to the file system that the file is stored on. And this could happen because there are bugs in the web server infrastructure.

Another way to attack grades.txt, or to discover the bits of grades.txt, would be to conduct a physical attack on the DRAM corresponding to Frans's computer and get the bits that way. Yet another way that's related is to get the backup copy of grades.txt from the file system that grades.txt is stored on. Every time you open up a file for editing, typically a temporary copy is stored in the same directory that the file is stored on. The permissions associated with this temporary copy are usually looser than the permissions associated with the original file. And yet another way would be to intercept the network

packets associated with the integer commands if grades.txt happens to be stored on a remote file system.

I've given you a bunch of ways, but there's so many more. So let me keep going. Nick could send Frans what's called a trojan text editor that is a binary that's been corrupted in a malicious way that has all the editor functionality that Frans is familiar with, but in addition, mails a copy of every file that Frans opens over to Nick. And that's the way Nick could get hold of a copy of grades.txt. Nick could steal the entire disk-- this is similar to accessing the disk blocks that I mentioned before-- and walk away with it, and look at the bits on the disk in his garage.

Two more ways and I'll stop. Suppose Frans happens to print out grades.txt and doesn't do a good job of shredding it, or just tosses it into a trash can. Nick could be an insider that walks away with a paper copy of grades.txt. While this may seem mundane, it's certainly happened. Nick could also conduct an impostor attack. He could call Frans, pretending to be sysadmin, and get hold of Frans's password, and simply become Frans, and look at all of Frans's files.

The point of this exercise was to give you a sense of the myriad ways that Nick could attack Frans and discover grades.txt, and I could keep going on and on. How do I know that I've thought of all the ways? I don't, and that is why security is a challenging problem, because it's a negative goal.

Security in IoT (Srinivas Devadas): Video 2

I want to talk to you about the diversity of threat models that face IoT system designers. Why are we interested in threat models? Threat models tell us what the power of an adversary is. Different IoT systems will have to deal with different sets of adversaries, and these adversaries are going to have different capabilities associated with them. So, in order to make progress, we need to figure out what a realistic adversary can do and design the system to block this realistic adversary.

It is clear from the Nick and Frans example that we cannot block an all-powerful adversary. What does a threat model actually look like? As I said before, it's going to tell us what the adversary can do and what the adversary cannot do. We could assume that the adversary controls some of the software in the computer, controls some of the users, in the sense that the adversary knows the user's passwords, but not all. The adversary typically also knows about bugs in deployed infrastructure or applications, because, a lot of the time, these are public and have been discovered in years past, put on websites, but these bugs haven't been fixed in deployed systems. So, an adversary has some power, some knowledge, but is not all-encompassing.

Moving on, in IoT systems, one is concerned not just with software attacks but also physical attacks and social engineering attacks. The physical attacks could be invasive attacks or noninvasive attacks. An example of an invasive attack is simply opening up a device, opening up the package corresponding to the chips in the device, and discovering secret keys. A noninvasive attack would monitor communication and discover private information that way. Social engineering attacks would simply correspond to corrupting an individual who has access to private information.

It's important to understand that many systems, including IoT systems, are typically compromised because of unrealistic assumptions about the attacker. By that, I mean that the designer thinks that the adversary is much weaker than he or she is. For example, the adversary is outside of the company firewall, and the firewall is going to block the adversary from getting any sensitive information. That the adversary does not know the legitimate users' passwords. And so these unrealistic exemptions cause vulnerabilities and misassumptions as to what the security of the system actually is.

I want to talk about how we could design systems that would be secure under particular threat models. What is our philosophy associated with designing secure systems? Cybersecurity is a property of the system, just like performance or energy is a property of the system. We can say that a system dissipates 20 watts of energy. We could say that a system is secure against a particular type of adversary. The important thing to note is that attackers take a holistic view of the system, just like you saw with the Stuxnet example, and discover a whole lot of detail about the system to craft their attack. And it's important, when we have such a diversity of threat models, for designers to take such a holistic view as well.

The current situation is that computer systems are designed to satisfy users and have a lot of features. Security is typically an afterthought. And so what happens is that the complexity of features gets us to the point where, because security is a second-class citizen, there are a whole lot of vulnerabilities associated with the implemented system. So the strategy that's espoused is that we should focus on building functionality into computer systems. And, when the computer systems are attacked, we discover the flaws in the systems-- or, rather, the attacker discovers the flaws for us-- and then we fix these flaws and move on. Unfortunately this happens over and over and repeatedly causes damage, because new flaws are continually discovered. This is what I call a "patch and pray" security mindset.

Can we do better than this? A holistic philosophy to our security is that we should think about security as a first-class design criterion and design security from the ground up as we design computer systems or systems associated with the Internet of Things. It's important to understand that a security property cannot be isolated to a particular component or layer. Just like the adversary has a system's wide architectural viewpoint, the designer has to think about security from the ground up and take a similar holistic view to the security of the entire system. Interfaces are particularly important to look at, since these are the places that most adversaries attack. If you can take a security "by default" approach to designing individual components and compose these components into a holistically-secure system, then, we've blocked the adversary from conducting successful attacks.

Security in IoT (Srini Devadas): Video 3 Part 1

Now that you appreciate the difficulty of security in the context of IoT systems, let me talk about how we could create systems that are secure against powerful adversaries. In particular, I'm going to talk about three defensive mechanisms of prevention, resilience, and detection and recovery. I'm going to give you examples of each of these mechanisms.

Prevention is about making the job of an adversary more difficult through design mechanisms. Resilience under attack is allowing for a system operation to continue even when an adversary is conducting an attack. One cannot prevent all attacks, so it's important for a system to be resilient under attack. And if a system can remain functional even when it's being attacked, you've, in effect, blocked the adversary.

Neither of these two cases may be possible. We may not be able to prevent a system from being attacked. We may not be able to continue operation when a system is being attacked. The system may, in fact, crash under an attack. In this case, it's important to understand exactly how the attack was conducted, to detect the avenue of the attack, and then recover from the attack.

Let me give you an example of a prevention mechanism. In particular, I'm going to describe how we could build systems that can prevent certain kinds of physical attacks. In IoT systems and other computing systems, we have functionality implemented in hardware, namely Integrated Circuits, or ICs.

Typically, you would like every integrated circuit to be unique in the sense that you'll want to be able to identify it as well as authenticate it. This is particularly important if the IC is running security software. For example, it's a processor running a security program.

Typically, we use cryptography to authenticate an integrated circuit. This requires the storage of a secret key inside the integrated circuit. And if the secret key is compromised, then we've got a serious problem, because anyone can spoof that integrated circuit and, in effect, become the owner or the user of that integrated circuit.

The way integrated circuits are authenticated is that you have a secret key stored inside the integrated circuit, and this corresponds to what's called a private key. And there's an associated public key associated with this private key that can be used by the authenticator or the verifier, as the entity may be called. Using a simple cryptographic protocol that corresponds to random numbers or nonsense, one can authenticate this integrated circuit in a cryptographically-secure manner.

The issue with this type of authentication, where secret keys are stored inside an integrated circuit, is that these secret keys are subject to invasive attacks that can extract these keys from the integrated circuit. Imagine if you had a whole host of integrated circuits that had the same secret key stored in them in an IoT system.

If this system was disseminated, or unsupervised, or deployed in a hostile location, anyone could walk away with one of these integrated circuits, open up the integrated circuit in his or her garage, discover the secret key, and be able to clone all of these circuits, and introduce false devices or fake devices in the field, thereby compromising the entire system.

So it's important that we create physical hardware that is resistant to this type of attack, we have prevention mechanisms that protect against these attacks. How can we generate secret keys in integrated circuits and protect them from physical attack? I'll describe one such mechanism in this part of the lecture.

It also may be the case that we cannot use standard cryptographic techniques like I showed you to authenticate a particular integrated circuit. This is because cryptography is, generally speaking, expensive. And resource-constrained devices, such as radio frequency identification tags, may not have the power to run these complex cryptographic algorithms. So we'd like to solve both of these problems in an inexpensive way in order to be able to build IoT systems efficiently and make them secure.

Security in IoT (Srinivas Devadas): Video 3 Part 2

I'm going to describe the prevention mechanism that is called physical unclonable functions that correspond to protecting integrated circuits from physical attacks to extract secret keys that are stored in the integrated circuit. In particular, physical unclonable functions extract secret keys from manufacturing variation as opposed to embedding secret keys inside the integrated circuit.

Just like no two human beings are created exactly identical, you can imagine that random process variation that builds to integrated circuits that have exactly the same design are going to produce implementations that are not exactly the same. There'll be minute differences in the topology or the physical structure of these two or more integrated circuits.

What is interesting is that these minute differences in the physical structure can actually be measured because they change the performance or the delay associated with integrated circuits. So in particular, if you have wires and logic gates on an integrated circuit, and two copies of these wires and logic gates have been built in two different integrated circuits, the delays associated with these two different copies will be different. If this delay can be measured, we can differentiate the first circuit from the other. And if we have hundreds of these, if all of the delays are unique, then we can differentiate each integrated circuit from every other circuit.

Here's a simple example of how manufacturing variation results in differing delays. What you see here are ring oscillators, a collection of ring oscillators, which simply correspond to a set of inverters that are hooked together in a chain. When you start up this circuit, corresponding to a ring oscillator, it starts oscillating. If there are an odd number of inverters in the chain, as you see on the slide, the frequency of oscillation is going to be unique for each of these different ring oscillators because of the manufacturing variation that I just told you about.

What you can now do is fabricate an array of these ring oscillators and compare pairs of frequencies of these ring oscillators to generate a single bit from each of these comparisons. If you fabricate an array of ring oscillators on a particular chip and compare pairs of ring oscillators, you can generate as many bits as you want from these comparisons. In effect, you can get an ID associated with one particular chip. And you can do the same thing for a different chip.

Because of the variation in frequencies of these ring oscillators, the first pair in a particular chip may be such that the top ring oscillator is greater in frequency than the bottom one. And you may produce a 1 for that particular chip. But in the second chip, it may actually be the reverse, where the top one is slower than the second one. If you fabricate enough ring oscillators, you're going to be able to get unique identifiers associated with each of these different chips. And you can use these unique identifiers as secret keys that are associated with each of these chips.

The crucial difference between this unique identifier and storing particular IDs is that these unique identifiers only exist when the chip is powered up. When you actually open up the chip or physically attack it when it's not powered up, all you see are the ring oscillators. And you don't see the bits. Contrast this with storing secret keys inside a disk or what is called electrically erasable programmable read-only memory, where there's physical differences in storing a 1 versus storing a 0.

When you compare ring oscillators, not only do you get a single bit corresponding to which ring oscillator was faster. If you, in fact, subtract the frequency of one ring oscillator from another prior to doing the comparison, you get a sense of how much faster a ring oscillator is in comparison to the other one. You can think of this as confidence information in the result of the comparison.

If you have a ring oscillator that is substantially faster than the other oscillator that it's being compared against, then typically, the comparison bit is going to be a stable bit, or what we call a confident bit. By that, I mean that if there is variation in the environment--for example, the temperature or the voltage-- there is going to be stability in the comparison because while the variation in the environment is going to change the frequencies of these two ring oscillators because the first ring oscillator is so much faster, it's going to remain faster regardless of the temperature or the voltage.

On the other hand, if you have two ring oscillators that are 100 megahertz apart, or one of them is a gigahertz and the other one is 1.1 gigahertz, these are much closer together in frequency. And it's possible the comparison bit would flip when the temperature got much higher or got much lower.

So not only can we compare frequencies and get bits associated with the comparisons, we can also get a sense of how stable these comparison bits are and derive what we call the confidence of a bit.

We conducted experiments on arrays of ring oscillators that were fabricated across multiple integrated circuits. This slide describes the summary of those experiments. What you see here on the x-axis is the distance between the unique IDs that are associated with generating bits from each different chip. 128 pairs of ring oscillators produced 128 bits for each chip. If I take two random chips and compare the vectors or the IDs associated with these two random chips, then they're roughly 64 apart, implying that we are essentially getting random identifiers out of these comparisons of ring oscillator frequencies. That's what you see in the middle of the slide.

Over to the left of the slide, you see an experiment where the same chip was asked to produce its unique identifier over and over. What you see is that the bits are not completely stable. You don't get the exact same identifier from the same chip over and over because of the environmental changes that I mentioned to you. There's maybe 10 bits out of the 128 that flip every time you ask for the chip to produce its 128-bit ID.

The important thing to note is that the level of error is much smaller than the difference between two different chips. What we call the inter-chip variation is what you see in the middle. And what we call the intra-chip variation is what you see on the left.

You can think of this as being a silicon biometric. Just like the DNA or the fingerprints of two different human beings are going to be quite different, you are going to have some variation associated with the fingerprint, depending on the sensor that was used to measure the fingerprint if you measure it on a particular day versus some other day. That's what corresponds to the noise or the intra-chip variation that you see on the left of the slide. And the difference between the fingerprints of two different individuals, that's the inter-chip of variation that you see in the middle of the slide.

This inter-chip variation causes some issues if you want to use the bits that are generated from the chip for cryptographic applications. In a cryptographic application, you need the secret key to be exactly the same when you used it to encrypt a piece of data and when you used it to decrypt a piece of data. If even a single bit is different, you want the result to be garbage. You want the secret key to be exactly identical in those two cases. So if we are going to use manufacturing variation to generate secret keys, we have to eliminate the intra-chip variation, or in effect, take all of these comparisons and ensure that they produce exactly the same result every time.

This is the notion of a fuzzy extractor. A fuzzy extractor, as its name indicates, takes a fuzzy source, be it ring oscillators or human biometrics, and extracts a stable secret key from this fuzzy source. A fuzzy extractor has two different aspects to it.

The first is generate, which takes the biometric source-- in this case, the ring oscillators, the comparison bits-- e as an input, and also takes s, which is the secret key that we'd like to create. And it's completely independently chosen from the comparison bits of the ring oscillators.

There's an algorithm called Gen, which is short for Generate, that produces what we called helper data, h, based on e and s. The idea is that in the repeat step, called Rep, we're going to be able to take a slightly fuzzy version of e, which may have some noise in it-- that's called e' here-- and use the helper data, h, along with e' to get back the exact secret key, s, that was originally chosen. So s is identical in both Gen and Rep, even though e and e' may have differences in particular bits.

The important thing to note from a security standpoint is that h was produced as a function of e and s in the Gen step. So from a mathematical perspective or a cryptographic perspective, it's quite possible that h would leak information about e or s. And of course, that would be a bad thing because we're going to store h in public form. And an adversary is going to be able to look at h. So we need to make some sort of argument corresponding to an information theoretic or an entropy argument or a computational argument that the knowledge of h does not allow the adversary to discover either e or e' or s.

Security in IoT (Srini Devadas): Video 3 Part 3

The way we're going to do this is by making a computational security argument associated with a hard problem that's called the Learning Parity with Noise problem.

Let me describe this problem to you. It's a fairly simple problem that corresponds to injecting noise into a linear system of equations. What you have here are that the a_i and s are n -bit vectors where s is an n -bit secret key and b_i and e_i are simply bits. s is secret, which is the secret that we want to protect. And a_i and b_i are going to be public, and e_i is the hidden noise that is going to be generated by our ring oscillators.

The Learning Parity with Noise problem simply says, that it's hard to discover s given the knowledge of a_i and b_i . While you could imagine inverting this system of equations and discovering s given b and a , it's hard to do that if there's a small amount of noise in this linear system corresponding to the e_i 's. Given Learning Parity with Noise, let me explain how the generate step works.

We're going to use the ring oscillator outputs as being the e_i 's. So we get a single bit for an e_1 , a single bit for an e_2 that is different, et cetera. s is chosen randomly and it's secret. We're going to create helper data that's associated with the b_i 's by taking public a_i 's and essentially computing the right-hand side here associated with $a_1 \cdot s + e_1$ producing b_1 , and so on and so forth.

So the b 's are going to be the helper data, the a 's are hard coded and they are part of the integrated circuit. They're the same for all systems, and the s is going to be generated randomly. And finally, as I said before, the e 's are going to be generated by the ring oscillators.

What is important to note is that an adversary who sees the b 's and the a 's and doesn't know the s 's or e 's cannot discover the s 's or the e 's because he or she would have to solve a Learning Parity with Noise problem that requires exponential time to solve using the best known algorithms.

Now, all we've done is encoded the helper data. We've produced the helper data using the Learning Parity with Noise problem. We need to be able to get back the exact s when we've powered up the circuit, maybe a day later, simply by taking the b_i 's from the outside that are the helper data and using them along with the a_i 's to generate the secret key s .

We want to regenerate the secret key now to be exactly the secret key that was originally encoded to produce the helper data corresponding to the b 's. We do have our ring oscillators. This is in effect what our fingerprint was. We do want to use our biometric ring oscillators to produce the secret key s . If you could get back exactly the ring oscillator bits that we had before, we could get back exactly the e_i 's, then we're done. We simply have to solve a linear system of equations because we have the exact e 's, we know the a 's, and we know the b 's, and we could solve this linear system to produce the s .

But you probably see the problem here. The issue, of course, is that the ring oscillators do have some noise in them. And we have a situation where the e's are not exactly the same as they were before. And these are in fact e primes, and a few of them are going to be in error.

The problem, of course, is that the Learning Parity with Noise problem, which I told you was a hard problem is the problem we need to solve in the repeat step now. Because the ei's are not exactly the same. So what have we accomplished here?

We have certainly encoded the secret key, s, in a secure way and created the helper data associated with it, but I haven't given you a rep algorithm that can get back the secret key. So I haven't quite solved the problem yet. The good news is that we're only one step away from solving this problem because we can go back to the notion of confidence that I spent some time on earlier when I described to you the ring oscillator Physical Unclonable Function.

In effect, there are going to be some of these ei's that are generated by a pair of ring oscillators that are far apart in frequency. These ring oscillators have confidence bit associated with them. Or the bits that are produced by the comparison are going to be extremely stable. They're going to be guaranteed to be exactly the same on each comparison because the frequencies associated with these ring oscillators are very far apart and the comparison is unaffected even by large changes in temperature.

So if I have a large area of ring oscillators, call it m being a thousand ring oscillators, is going to be an overwhelming probability that 128 of these ring oscillators are going to produce extremely stable bits when they're compared against each other. These 128 bits can be discovered simply by looking at the confidence information associated with each of the bits that is in fact generated by the subtraction of the frequencies.

So I can in effect find a subset of ei primes that are guaranteed to be the same as the ei's. I can select 128 of these equations and solve now for the 128-bit s because all I need are 128 correct equations to obtain 128-bit secret key. So now I'm in a situation where repeat is going to work with overwhelming probability given the stability of the selected bits.

Just to go back to ring oscillators, and to elaborate on what I just described to you, what we do is we use the confidence information associated with the subtraction of the frequencies to zoom in on the bits that are guaranteed to be the same across different invocations or different comparisons.

Once we have discovered this stable bits, we can get exactly those 128 equations that will get us the same key that was created during the gen step and encode it into the helper data corresponding to the bi's. The theoretical result that you have is something that tells you about the reliability of this process as well as the security of this process.

On the reliability side, if we make m large enough, that it's order n squared, it grows as n square, we're guaranteed with overwhelming probability that we're going to get the exact

same secret key back through one Gaussian Elimination that takes order n cubed time. And so think of n as being 128 here and m as being n squared, which would be 10,000 or so.

That is a reliability side of things. On the security side, remember, that the adversary does not have access to the confidence information, does not have access to the e_i 's. The adversary only sees the a 's and the b 's, so the adversary has to do exponential work in order to solve the Learning Parity with Noise problem in order to discover the secret key s , which is that exact situation we want to be in from a security standpoint.

Exponential time to solve the problem is intractable for all modern computing systems, so we are in good shape here. Let me close by stitching things together with respect to this particular prevention mechanism. and IoT systems. What we've created here is a physical unclonable system that can generate secret keys that only exist when the device is powered up. You can now generate secret keys without having to embed them because the secret keys come out of manufacturing radiation not from specific programming that a trusted programmer needed to do for a particular chip. And if the trusted programmers is not trusted, then you expose the secret keys to anybody who would bribe the programmer and discover them.

Physical Unclonable Functions or PUFs for short can enable secure low cost authentication in resource constrained system. And the PUF can generate a secret key through extracting the secret key from manufacturing radiation as opposed to having to embed a secret key in a secure location into each and every chip.

This secret key only exists on power up, and is physically much more secure than non-module keys that are stored in flash, or on disk, or on EPROM.

Security in IoT (Srini Devadas): Video 4 Part 1

I'm going to describe a second mechanism that corresponds to resilience under attack. As I said before, we may not be able to prevent all possible attacks. So what happens when a computer system or an IoT system is actually attacked? We'd like to continue working, even under attack.

The trusted computing base of a computer system corresponds to all of the software and the hardware that needs to be trusted to work correctly and not be malicious in order for the system to be secure. Unfortunately, modern computing systems have huge trusted computing bases. It's not unsurprising to see computing systems that have 20 million lines of code corresponding to their trusted computing base. And when it comes to IoT systems, the amount of hardware and software that needs to be trusted is even larger.

What we'd like to do is shrink the trusted computing base. We'd love to be able to say things like we only trust this piece of hardware to protect its secret keys. And we don't trust any of the software in the system or any of the extraneous hardware, for example, memory systems or disk systems.

One way of doing this, to shrink the trusted computing base, is to use encrypted computation. This is going to allow us to be protected against attacks because adversaries that discover data cannot do anything with it because the data is encrypted. The notion of encrypted computation is that I'm going to delegate the processing of my data to an untrusted server without giving it access.

What do I mean by that? Suppose I could encrypt my data and separate the processing of data from access to the data. I'm going to send my data encrypted to the untrusted server. The untrusted server is going to compute on encrypted data. It's going to send me back an encrypted result. When I decrypt the result, I get the answer that I want as if the computation occurred on the unencrypted data, and the same sequence of additions and subtractions happened and produced the correct result.

This notion of computation under encryption is a powerful notion. And I'm going to use an analogy to describe it to you in perhaps a more compelling way. Let's say Alice wants to buy a beautiful ring for herself. Not only that, she wants to design this ring. She is going to hire jewelry workers to create this ring for her and give them raw materials to do this. But there's a problem here. The problem is one of theft. The jewelry workers could create the ring for her and just walk away with the ring. How could she protect against this scenario?

Alice could create a locked glove box and put her raw materials inside the locked glove box. Alice puts the raw materials in a locked glove box. The jewelry workers are going to put their hands into the locked glove box, work on the raw materials, and create a ring, except that they have no idea that they're even creating a ring. It's only Alice that knows that they're creating a ring for her. The jewelry workers, after they have finished their task, are going to take their hands out of the locked glove box and walk away. Alice will

presumably pay them for their work. But now Alice is going to be able to open up the locked glove box in private and take out her beautiful ring and enjoy it.

Given this jewelry example, let me tell you exactly what happens from a mathematical standpoint. The analogy here is that encrypting is putting raw materials into the locked glove box. So the raw materials correspond to sensitive data that's associated with Alice's DNA, for example.

Decrypting is taking things out of the box. As I mentioned, the jewelers have no idea that they're building a ring. They simply produce an encrypted result in the mathematical domain. Alice is able to take the encrypted results and decrypt it to obtain her diagnosis. The computation is the process of assembling the jewelry, and this corresponds to computing on encrypted data. We need particular mathematical structures corresponding to the encryption and decryption algorithms to ensure that the computation on the encrypted data, to produce an encrypted result, produces exactly the same result as if Alice had computed on the original sensitive data using standard operations.

Given this analogy, encrypted computation corresponds to requiring this mathematical structure that has the technical term of fully homomorphic encryption. Fully homomorphic encryption ensures that the decryption of the encrypted result is the correct result. It also ensures that it's hard, without knowing the secret key, to be able to decrypt the encrypted data, or the result. This is a wonderful notion that would solve a whole host of security problems, including those that we encounter in IoT systems.

Unfortunately, there's a problem with the current state of the art in FHE, or fully homomorphic encryption. FHE techniques are really, really slow. The same computation that takes a second to run on a modern computer, if it's run in the encrypted domain, would take a billion seconds or more. So we cannot use FHE for complicated computations.

The question is, can we speed things up using trusted hardware? And this is what I'd like to talk about next.

Security in IoT (Srini Devadas): Video 4 Part 2

We could try to use tamper-resistant hardware, a secure processor that stores a physically secure key in it, to solve the encrypted computation problem. In this case, we have a secure processor that we're going to assume is inviolate, in the sense that you cannot look inside of it. And it's going to be able to perform operations by decrypting the data inside of it and running computations on the decrypted data. The decrypted data is never exposed outside of the processor. This is what tamper-resistant hardware can give you.

The idea here is that performance would be almost identical to running things in the unencrypted domain, because, in fact, you're going to be decrypting the data internally and running standard operations like addition and subtraction. You're not going to see the overhead or the slowdown that the FHE approaches have.

Unfortunately, tamper-resistant hardware has its limitations. While we can realistically assume that we cannot open up the hardware chip and discover the computations inside of it, partly through the use of physical [INAUDIBLE] function technology, it is the case that the adversary can monitor the pins of the chip. It's easy for an adversary to tap the memory bus, which is exposed, or to tap the pins of the chip, and discover things like how much power is being dissipated, or, even more realistically, discover what the memory addresses are corresponding to the computation being carried out.

Examples of these attacks corresponding to tapping the memory bus are tapping the I/O channel or the address bus that corresponds to connecting the processor to memory. A simple example of how leakage through the address channel can expose secret information is shown here on this line.

Imagine that you have a simple computation, where based on a secret value corresponding to x , the computation is going to access one particular piece of data or another piece of data. As you see here, if x is 0, I'm going to access the i th index of the array i , and i is changing. On the other hand, if x is 1, I'm going to access a 0.

Depending on the value of x , if an adversary is monitoring the address channel, he or she is going to see different addresses corresponding to 0, 1, 2, et cetera, when x is 0. We're just seeing the same address over and over if x is 1. This is immediately going to tell the adversary what the secret bit x is without having to break apart the chip, and just looking at the memory address bus.

We obviously have to protect against this type of attack if we want security in our IoT system. The SM processor is state-of-the-art secure processor that protects against this type of attack corresponding to monitoring the memory address bus. In particular, it makes the sequence of memory addresses for any particular computation indistinguishable from any other computation. It gives a cryptographic guarantee that there's no leakage of sensitive information through the memory address channel. It also ensures that the timing of these memory address requests is such that there's no leakage of private information corresponding to the timing, as well. To summarize, the SM

processor is going to eliminate all leakage from its address pins when it's accessing external untrusted memory.

What you see here is the chip responding to the SM processor. And what you see here is untrusted external memory corresponding to DRAM in a computer system. The adversary could be tapping the pins associated with SM. We can assume that the adversary cannot look inside of SM.

In order to block all leakage corresponding to the pins, we're going to have to do two different things. We are going to ensure that the memory addresses corresponding to the DRAM requests are office-gated, using a cryptographic primitive that is called oblivious RAM. Rather than making a single address request, the SM processor, for each request, makes a sequence of addresses that is indistinguishable from other addresses that the processor makes, regardless of the computation that it's carrying out.

The second thing that it's going to do is it's going to run for a fixed amount of time and make periodic address requests to the memory, so the timing channel is blocked as well.

With these two features, the processor guarantees cryptographic security associated with memory address requests.

The SM processor would be the processor of choice in IoT systems where memory address pattern attacks or attacks on chip pins are realistic, and can be carried out by a motivated adversary.

Security in IoT (Srini Devadas): Video 5

I'm going to describe a third mechanism to produce secure IoT systems. It may be the case that we can prevent sudden attacks from happening, but we cannot prevent all attacks from happening. In that case, we could imagine that we could try to remain functional by having mechanisms such as encrypted computation where loss of data is not an issue.

But it may be the case that we actually suffer from a particular attack and the system actually crashes. In this case, what we have to do is detect and diagnose this attack and recover from it as fast as possible. I'm going to talk about mechanisms that do just that.

Sometimes, one is not just concerned about privacy leakage, as we saw in some of our examples, but we're concerned with the correctness of the computation. It may be the case that the adversary simply wants to corrupt the data and create malicious computations on the device that end up with the user making wrong decisions. In order to ensure integrity, we have to ensure integrity of storage-- that is integrity of data-- as well as the integrity of computation.

The integrity of storage is an easier problem, relatively speaking. One can use redundancy codes to guarantee that any corruption of data isn't detected. Integrity of computation is a much harder problem. This is because computations can be corrupted in many different ways. An error could be pretty subtle, or it could have a catastrophic effect. We have to ensure that any corruption is detected through our security mechanisms.

The classic way of ensuring integrity of computation is through parallel or redundant computations. If we can compute a particular result in two different ways and we can check that both of these computations resulted in exactly the same result, then we have some confidence in that the computation was carried out correctly. We can obviously increase the number of ways to increase this confidence.

The challenge in this approach is to keep the performance overhead manageable. We don't want to do too much redundant computation and suffer orders of magnitude overhead. The key idea here is that we'd like to figure out what computations are likely to be carried out correctly, that are unlikely to be attacked, and what computations are less likely to be carried out correctly.

We would only like to introduce redundancy for the less likely computations. By that, I mean the computations that are less likely to be carried out correctly. The likelihood of a computation being carried out correctly or the likelihood of data being uncorrupted is a function of the source of that data. Data from an external source, for example, has less likelihood of being uncorrupted versus data from a trusted source or internally-generated data.

The mechanism that we're going to use for redundant computation or efficient redundant computation is an architectural mechanism that is going to track the confidence of the computation and the confidence of the data that we're computing on. By the confidence of the computation, I simply refer to our confidence that the computation is being carried out correctly, or in the case of data, our trust in that the data has been uncorrupted.

Architectural mechanisms given confidence of input data can compute confidence of intermediate results and the confidence of the final result through a process of tracking these measures throughout the computational process. These mechanisms can be built into processor hardware, and the processor can carry out these computations in an efficient way.

This notion of confidence tracking has been used in the case of preventing buffer overflow attacks in modern computing systems. In IoT systems, we can have a much wider variety of attacks, both physical and software attacks, and the level of sophistication required in tracking confidence is higher.

Nevertheless, one can imagine architecting a processor that tracks the confidence of data throughout the computational process and re-performs the computation-- that is, performs redundant computation if the confidence drops below a certain level and raises the confidence level up through the use of these redundant computations. Such processors do not exist today, but are currently under active development.

Let me summarize what I've told you. I started out telling you why security was a hard problem and talked about the diversity of threat models. I moved on to describing defensive strategies or mechanisms that could be used to create secure IoT systems. Such strategies fell into the categories of prevention, resilience under attack, and detection and recovery.

I gave you one example of each such defensive strategy. There are many other examples. And typically, these examples correspond to different layers of abstraction or correspond to different layers of software and hardware in an IoT system. To build a secure system may require such mechanisms at all layers of abstraction-- the compiler, operating system, the application, and the hardware. I hope you've enjoyed this lecture on security issues in the Internet of Things. Thank you.

HCI in an IoT World (Jim Glass): Video 1

I'm Jim Glass, and welcome to the next module of the Internet of Things, which is entitled Human Computer Interaction in an IOT world. Before we start, let me tell you a little bit about myself. I'm a senior research scientist at MIT, and I had a research group in the Computer Science and Artificial Intelligence Lab that works on advanced speech and language processing technology.

I worked in this area for over 25 years. And what I hope to do, over the course of this module, is share my expertise and perspective with you. So over the course of this module, we're going to have several different segments. In this first segment, we're going to introduce the concept of HCI and IOT. And then I'll move on and talk about some actual prototypes to give you a sense of how the technology works.

Then, I'll go behind the scenes, cover the underlying technology, talk about multimodality for HCI, and then talk about some of the computational issues that we might want to consider in IOT world. And finally, I'll end up talking about some of the challenges for future devices.

First, let's talk about human computer interaction, in general, try to place that. Generally speaking, HCI focuses on the interface between you and me and our stuff on the internet, whatever it might be. Movies, calendar information, other things you want to search for-- how do we get at it?

You know, historically, there's been the interface with a desktop computer. We're all very familiar with the Windows, the icons, the mouse, pointing at things, the wimp interface. With the advent of smartphones, now it's slightly different. The graphic user interface on the phone, you use your finger to gesture and click on things.

And now we're starting to move to other devices, like smart watch. The displays are getting smaller, you can tell, and how you're going to get access to your information? These are the types of things that people who were in HCI worry about.

So, in particular, for this module, we want to think about the use of HCI in an IOT world. How are we going to communicate with all of these network devices of different sizes, big and small? If you think about it, the conventional interface, on the desktop with a keyboard and mouse, probably isn't going to work. It's great in an office environment, and it'll probably stay that way, but in other scenarios it's going to change.

And what I'd like to persuade you, now, is that I think speech is going to be a great modality for us to interact with our devices. And let me explain why. Over the last five years, I would say, speech is reached a tipping point in our society. It's starting to get out there in the public with things like Siri, Android devices, Cortana-- the Microsoft-- and now things like Amazon Echo.

Everybody's starting to use speech in their devices, and they're seeing the potential power of it. In fact, it's so great, people want to use speech everywhere. Whether they're at home, whether they're at work, whether they're in between, or somewhere else, out with their friends, it's just so natural for people to use speech.

And by the way, when I say speech, I mean a lot of things. To humans, when we say, oh, let's speak, it really means not exactly what I say, it's what I mean. So we really need to involve understanding when we're interacting with our devices. We need to worry about, maybe the machine might need to interact with us, have some kind of dialogue. And it might even need to generate some kind of speech, like things like Siri does.

So these are all the underlying technologies we have to worry about when we think about speech interfaces with our devices. So why speech? Well, we all speak. Our entire lives, we speak. It's very natural. We don't need to be trained specially to use some kind of new interface. Just talk.

It's very flexible. My hands and eyes to be busy doing other things, and I can still talk. That's great for many different scenarios. We talk very efficiently. You can do up to 200 words. Most people can't type that faster or text that fast-- although my teenagers are pretty good at it.

And it's very economical. We can transmit and receive speech very efficiently, economically. In particular, speech, I think, is ideal when the information space we want to navigate is very complicated, just ask questions and have the computer understand what we're talking about.

Another scenario that's useful for speech is when the users are naive. Perhaps experts, for some particular interface, could do something very efficiently. But the average person, like you or me, speaking is just easier for us. And finally, as devices get really small, those keyboards are going to get tiny. So just talking to it, I think, makes a lot of sense in these kinds of scenarios.

So when people think about speech technology, the first thing that probably comes to mind is the notion of taking a speech waveform and figuring out what words are said. And that's known as speech recognition. But there's other things that you have to worry about, too. The opposite direction, for example, is speech synthesis, or text to speech, where we have an underlying text representation and we want to generate natural and intelligible-sounding speech.

But we have to go beyond the text representation. And that's where underlying meaning extraction, doing natural language processing to figure out what the concepts you're talking about, what's the relationship between these concepts to actually do something appropriate.

So that goes for understanding, generation, and maybe the machine has to manage the interaction as well, and that's where dialogue management comes in. So if we think

about, what are the technologies that are needed for futuristic advanced speech interfaces, it's really the notion that the devices can interact with people using the mode of conversation.

That means they have to understand the verbal input-- speech recognition, language understanding-- and they have to understand in the larger context of what's been said and maybe even what's been going on in the general environment. They have to worry about generating response on the output side, communicating back to the user because interaction is both ways. And again, engage in a dialogue with the user.

Now, this brings me to the end of the first segment, the introductory segment of this module. What I'm going to do next is go on and show you some actual prototypes where you can see people interacting with these systems.

HCI in an IoT World (Jim Glass): Video 2

Welcome back. In this segment, we're going to actually show some real prototypes so you can get a sense of how people can interact with their devices. We have actually been developing speech interfaces for a long time. And part of the motivation for developing interfaces is interaction between humans and computers is actually a chicken and egg problem.

In order to make the technology work well, you need to have people interacting with a prototype so you can collect the data, because all of the underlying technology is data driven. You need data. You need speech. You need language. You need to see how people interact with the machine.

But in order to have people interact with a system, you need to make a system. So first, you make a baby system. You get people interacting with it. Then you can improve the technology over time as you collect more data. I've actually been talking with computers, if you can believe it, since 1989. I've been asking for where's the nearest Chinese restaurant for a long time.

In the early days, we just had local machines that people would interact with. Remember, the web wasn't even invented then. Starting around the mid 1990s, we started deploying technology on the web. And in the 2000 era, we sort of could deploy it on a large scale on the web.

But in the late 1990s, we started making our systems publicly available on the old-fashioned telephone system. We'd have toll free numbers. People could call in. What's the weather going to be like tomorrow? Things like that. Very popular. But it's much more compelling, of course, when you have visual information you can convey to the user as well.

So what I'm going to do is show you a couple of systems. The first one is sort of a air travel scenario. I should say that when we deploy these prototypes, we've always tried to pick scenarios that are common for people. Everyone worries about the weather. People maybe want to find about restaurants. People need to go places. Things like that.

So the travel information scenario is a system that you can interact, speaking naturally. Asking about flights. And you can browse them, for example, in this case, on your phone. So let's play the video.

I'd like to book a flight from Boston to San Francisco leaving Friday afternoon. Can you show me the JetBlue flights? Are there any nonstop flights? Actually, I'd rather leave in the morning of October 5th. Can I return on the afternoon of October 9th?

As you could see from the video, there is a little record button he was tapping to talk on the screen. It would show the recognition result. It would have little boxes around the key concepts in the understanding. And then it would show flights. He could actually have

clicked on those to expand them if he wanted to. And it would show a little balloon showing the system response of the user.

So that's one example. Let me show you another system that focuses on nutrition. This is actually one of our most recent prototypes that we're working on, and it's related to health. A lot of people care about logging the information-- the food they eat.

And some of the existing interfaces out there are kind of tough to use. They're tedious. People stop using them over time. And in our collaboration with some nutritionists at Tufts University, the idea is that maybe if people could just talk about what they're eating, it would be easier. So let's play the video.

For breakfast I had a bowl of Quaker Oats, followed by a Chiquita banana, and a glass of juice. It was orange juice. For lunch I had a cheeseburger from McDonald's with an ice cream sundae. For dinner I ate a plate of pasta, as well as a Greek salad with olives and feta cheese. I had five olives.

Again, you can see that there's a microphone that they click to record. This is now using HTML5 web protocol. There is a server that's doing the speech recognition, another that's doing the understanding. So you can see the little boxes where the concepts are highlighted.

And then we take that, and we go off to the USDA database or other databases, get the nutritional information, show that to the user, and they can then edit it manually, or they can follow up with another spoken query.

So those are examples of prototypes. And that brings me to the end of this segment. What I'm going to show you next is we'll go under the covers a little bit. And I'll talk about some of the technologies that we're dealing with.

HCI in an IoT World (Jim Glass): Video 3

Welcome back. In this segment, we're going to talk about some of the underlying technologies that are needed for these advanced speech interfaces.

But let's start with the system architecture for that nutrition system that we just looked at. On this slide, we have the three major components of the nutrition system. Of course, over on the left side we have the interface itself on the device-- the web browser, or the tablet, or whatever you have, where you get the user input. Maybe it's speech. Maybe it's clicking on the interface, and you show the output and you log things.

Now, behind the scenes, there's a couple of different servers. One of them is doing all of the speech and language processing. And this is where I'm going to go into more depth, where you need to figure out what they said, what the concepts were, and what is it that they're asking about.

Once you've figured that out, you can then actually go to your database and look up the information and do the actual information access. And this is the types of things that go on in this other area. I'm not going to address that in this module at all, but it's an important component, getting real information for people to look up. I'm going to focus on the language technology, so let's talk about that right now.

If you think about somebody interacting with a system, here's our little person in green with a microphone, and he or she is going to say something. And of course, first you send the speech to a speech recognizer to try and figure out what words are being said. Now, you might not need to just do the very top choice. You might have a bunch of alternatives.

But you get that, and then you feed it to another component, which has to do what we call language understanding, figuring out the important concepts and their relationships to each other so that you can produce some kind of meaning representation of what they said.

So all the different ways you might say, what's the weather going to be like in Boston tomorrow, is it going to be nice in Boston tomorrow, all the different ways you could speak it, the meaning representation is ultimately the same. In fact, I'll get to this in the future, but if you said it in a different language, the meaning representation might be the same.

Once you have the meaning representation, then you can go to your database, and you can show information to the user. You might need to have an interaction. If they didn't give you enough information, you might need to say, can you tell me something X, or if there's too much. This is where dialogue can potentially play a role.

The other thing I should say is there's this notion of contextual resolution or discourse context. Understanding often has to be done in the context of what's been said before, and

so that plays an important role. And finally, in addition to graphs and tables, you might actually want to say something back to the user, generate a summary of the information that they can see or speak to them. In some scenarios where there's no display, you might just only be able to speak to them.

So these are all the types of technologies that we have to worry about when we put together an advanced speech interface. So let's go through one by one, and I'll tell you a little bit about them.

So speech recognition. Actually, it's a very complicated process, takes a lot of computation. Although the technology has continually improved over the last 40 years, if you can believe it. Even though Siri just came out in 2011 or whatever it was, people have been working on this a long time.

Fundamentally, speech recognition you can sort of think as a search problem. Over here on the left, we have a speech signal. This person, believe it or not, actually said, 9 Davis Square in Somerville. And one of the things you'll notice is there's no boundaries between the words in the waveform. It's a continuous sequence of sounds. Even though the words pop into our head as discrete symbols, it's continuous, and no one's telling us where anything is.

So fundamentally, the speech recognizer has to consider that every word in its vocabulary-- and vocabularies can be up to a million words-- could start at any point in time. So it's proceeding through this waveform considering all possible candidates at any point in time, and it has to figure out overall the most likely sequence of words.

It's a very big search problem. Sort of like chess computers looking ahead-- what are all the moves the person might make? What are all the countermoves I might make, and do that several levels deep. Big search problem. Computers kind of doing the same type of thing here, considering all these candidates and trying to figure out the best one.

Now, what goes into the search? Well, there's some representation in terms of time and energy at different frequencies that's typically used. There's constraints. For example, the language model tells us about the likely words a person might use. If it's open-ended, it'll be very general.

But languages tend to have a particular order. Like in English, it's subject, verb, object. If I didn't do that, I might sound like Yoda if I was talking. What's the weather like in Boston? Boston weather like what is, or I don't know.

We have a lexicon, where we talk about the words and the vocabulary and what their pronunciation might be. If you look at the teeny little sequence of sounds after each word, that's telling you the canonical pronunciation for how it's spoken. Of course, in fluent, conversational speech, people typically don't pronounce things the canonical way, and that's another thing that makes speech recognition hard.

Finally, another constraint we have are the acoustic models, where we have some model of what different sounds look like in different contexts for the language, and those will be language-specific. And actually, the sounds that I produce are different from the sounds that you produce. So it's a difficult problem modeling this for a general population.

So all of these components go together, mix together in the search, and we figure out the most likely sequence of words, or maybe a graph, or a network of close competitors. And that's the type of thing that goes on in speech recognition.

The output then gets passed to the language understanding component, where we try to figure out the concepts. Currently, again, we're using machine learning methods to try and figure out a sequence of labels for the word. So here in the food domain, I had a bowl of Kellogg's Frosted Flakes. It goes through word by word and tries to figure out whether something, in this case, is a quantity, a brand, a description, or a food, and it figures out the most likely labeling sequence.

In addition to labeling individual words, we also need to perhaps group them together to figure out which concepts go together. So for example, I had a bowl of cereal. I didn't have a bowl of milk. I had two cups of milk. So there's a whole other process where you figure out, OK, this bowl goes with the cereal, and the cups goes with the milk. And you can do that, and you can then produce a meaning representation and you can look up information.

Now, I've mentioned a couple of times that understanding has to be done in the larger context of what's being said. So if somebody says, it was Greek yogurt. So in this scenario up here on the upper right, I had a cup of yogurt and a strip of bacon. And then they want to enhance the description of the yogurt. It was Greek yogurt. Well, you need to understand the word Greek in terms of what was said before.

I had some milk. And then they say, well, actually it was three cups of milk. That's actually a lot of milk to have at any one time, but apparently that's what they did. And the system should be able to understand what the user said and incorporate it correctly, either by modifying the yogurt or modifying, in this case, the quantity of milk that they ate. So those are types of things that you have to do for contextual understanding.

The next component that's part of a spoken dialogue system is the dialogue management component, and it's actually a very challenging problem. In fact, most dialogue managers have not made it out to the general public because it's hard to do it in a general way. But let me give you some examples. One area where the machine might want to interact with a user if the input is ambiguous.

So here's a scenario going back to the travel prototype. Somebody's flying from Boston to San Francisco. Maybe the machine wasn't sure exactly what they said. Is it Boston or Austin? So they need to clarify that. And once that's straightened out, they say, oh, gee, actually I need to know when you want to travel. So they're gathering additional

information. So these types of things often go on before you can actually go look up information.

Another area where dialogue is important is relaying information to the user on the output side. Maybe, in this case, you found a whole pile of things that satisfy the user's request, and you might want to narrow it down a little bit. So you might solicit a little more input from the user. So these are just two very small examples of the role of dialogue management in the interface.

Finally, there's speech generation. Do you want to say something to the user or show them a summary? Oftentimes you can show them a big table of information like this. But you might actually want to use language generation to generate a description of what they're seeing.

And one of the advantages of that is it really shows the user that the computer understood what they were saying. It's an implicit form of confirmation of, OK, yeah, they got all the concepts correct, so the list of information here is what I wanted. And of course, you might want to speak that as well using speech synthesis.

So these are all of the underlying technologies, and that brings me to the end of this segment. Next what I'm going to do is think about speech in a larger context of different modality inputs.

HCI in an IoT World (Jim Glass): Video 4

Let's start talking about multi-modal interaction. Now, I'm a speech guy. I love speech, worked on it all these years. But I'm the first to admit that there are great places for speech. And there's probably not so great places for speech.

Having everybody dictating their notes in a classroom, probably not so great. Telling the computer my password at an ATM, probably not so great. So speech has its place. But there's an important role for other modalities, as well.

Now, in the last segment we talked about technologies involved in spoken interaction. And that's what you have here. But you know what. Different modalities are great. They give the user flexibility. They can pick whichever modality they want under their particular circumstances and their personal preferences. That's the way it should be.

It's efficient. Sometimes trying to describe something on a map using speech is really tough. It's just easier to maybe draw a circle around the region that you're interested in.

Also, having multi-modal inputs can potentially be very robust. They complement each other. And they're also consistent to the larger message or the larger meaning of what the user is trying to do.

So what are some of the modalities that you might consider and that we've looked at? Well, there's handwriting, of course, pen-based interaction. More generally, there's gesture, things like the Connect, where you're pointing around at things. And that would be useful for bigger environments.

You can also use visual information, having the computer look at the person. What are their eyes doing? Where are they engaged? What is their mouth doing? In a noisy environment, when we're in a party or something, it's very noisy, we often focus on the person's face that we're listening to. We're using cues from their mouth to provide additional input to what we're hearing to do more robust understanding.

So humans do it. Machines should probably do it, too.

So I'm going to show you a couple of examples of multi-modal systems. This is one that we worked on that does sort of navigation in city areas. It's a multi-modal interface. You can talk to it. You can use a pen. You can type to it. You can select things on the menu. It's in a web-based environment, so it's accessible to people. And we actually incorporated other technologies into it.

Multi-modal, as I mentioned, you'll see, you can actually imagine personalizing it. Every interface should be personalized to the person. If it's your own device and with your little fitbit, we should know it's you actually. And we can use that to our advantage. And I'll say something more about that in the future.

It should be customizable. I should be able to give names to things that I want. Maybe I should name the devices that I want, give them particular names. We have Siri. We have Cortana. We have Alexa. Maybe people want to call their toaster their own particular name that they can talk to it. I don't know. But we should be able to do that.

And finally, as I have said a couple of times, it should be contextual. The devices should know the larger context of what's going on to help constrain the understanding and do a better job.

So let's take a look at one of these prototypes.

PROFESSOR 2: I'm going to demonstrate the City Browser, which is a web-based application that allows anybody with an internet connection and a microphone to obtain information using speech and gesture about restaurants, subway stations, and museums in various metropolitan areas. Imagine that I'm visiting Boston. And I'd like to spend the day visiting different museums using public transportation.

I'm going to use City Browser to find out about what museums are in the city and to find public transportation options near those museums. I'm also going to find a restaurant near one of the museums.

Show me museums in Boston.

COMPUTER-GENERATED VOICE: There are 15 museums in Boston. They are found mostly on North Square, Joy Street, Science Park, and Fenway.

PROFESSOR 2: So I can see where the museums are located. And I can click on each one to find out more information. I can also locate a specific one.

Where is the Paul Revere House?

COMPUTER-GENERATED VOICE: The Paul Revere House is located at 19 North Square in Boston. The phone number is 617-523-2338.

PROFESSOR 2: Are there any subway stations near this museum?

COMPUTER-GENERATED VOICE: There are three stations near the Paul Revere House-- Haymarket, Aquarium, and North Station.

PROFESSOR 2: Zoom out, please.

COMPUTER-GENERATED VOICE: OK.

PROFESSOR 2: Are there any recommended Italian restaurants here?

COMPUTER-GENERATED VOICE: There are 52 recommended Italian restaurants in the region you specified. The particular neighborhoods are North End and Government Center.

PROFESSOR 2: Can you tell me the hours of Antico Forno?

COMPUTER-GENERATED VOICE: Here are the hours for Antico Forno. Daily, 11:30 AM to 10:00 PM.

PROFESSOR 2: The City Browser works in several different metropolitan areas by loading in different vocabulary dynamically. For instance, it will load in a new set of street names and city names as well as restaurant names and subway stations if I switch to a different metropolitan area.

Imagine that I'm traveling to the San Francisco area for a business meeting in Palo Alto. I'm going to find the office building where the meeting is. And I'm going to find a restaurant nearby to have lunch at.

Show me 165 University Avenue in Palo Alto.

COMPUTER-GENERATED VOICE: Here is 165 University Avenue in Palo Alto.

PROFESSOR 2: Once I've told the system an address, I can give it a name so that I can refer to that address later.

I'd like to add a landmark here.

COMPUTER-GENERATED VOICE: OK.

PROFESSOR 2: I can now type in whatever I want to name this address. Here, I'm going to put in the name of the company where I'm going.

COMPUTER-GENERATED VOICE: OK. I have added a landmark named New Web Company.

PROFESSOR 2: Are there any Chinese restaurants near the New Web Company?

COMPUTER-GENERATED VOICE: There are 10 Chinese restaurants near New Web Company. Many of them are on El Camino Real.

PROFESSOR 2: Tell me more about Windys Chinese Restaurant.

COMPUTER-GENERATED VOICE: Windys Chinese Restaurant is located at 168 University Avenue in Palo Alto. It serves Chinese food. The phone number is 650-325-3188.

PROFESSOR 2: Great. I'm done.

PROFESSOR 1: So you can see in this movie there was a lot of different technologies going on. This was on a tablet. But you could imagine deploying these technologies everywhere.

This brings me to the end of this segment. And in the next segment, we're going to talk about combining different modalities in parallel, in particular audio and visual information because they're going at you at the same time. They both convey information about different aspects of what's going on.

HCI in an IoT World (Jim Glass): Video 5

So in the last segment, we talked about how to combine different modalities-- speaking, gesturing together. But there's a different kind of modality where things are coming on in parallel, and that's what I want to address in this segment that I call multi-modal symbiosis.

If you think about the acoustic signal recording sound and the visual signal recording images, they actually both tell you interesting things about the environment and the person in particular. So for example, if you had microphone arrays or cameras, that can help you figure out where the person actually is. Maybe you have multiple people, and you want to focus on the one person who's talking. You can use both sources of information to help you do that.

Maybe you want to figure out who the person is. There's actually information in the sound of your voice conveying who you are and, of course, doing face recognition, but things like gait recognition even tell you something about the individual. All of these things can be combined to help figure out personal identity, and that can be useful in different scenarios.

Of course, there is the process of figuring out what they said or the linguistic message of the words, and that's where you'd have speech recognition, maybe with processing lip movement to do robust speech recognition and noise. And finally, there's an important aspect of speech, and this is how speech differentiates itself from email. There's the emotional content in the signal, and extracting that information can be very useful as well. And you can do that both from the acoustic signal, and you can also do it from the expression on a person's face. So that we call paralinguistics, nonlinguistic information, and that's an important source of information potentially for intelligent user interfaces.

So when you combine all these different technologies, you can see that different modalities provide useful, complementary, consistent information that will let you do a better job in each of these different kinds of tasks. So let me give you one particular example, and that's multi-modal person verification, where you might want to identify who a person is using sound and vision.

We did an experiment awhile ago using state-of-the-art technology for both speech and face, and what we found when we put them together-- you reduce the error rate by a factor of 10. Each of them individually had about a 2% what we call equal error rate of falsely identifying somebody, but when you combine them together, we got 0.2%, so that just shows you the power.

And you can put other things in here as well-- like I mentioned, the way somebody is walking. These are all sort of noninvasive means to try and identify somebody, and it can be useful in different scenarios. Maybe your doorknob going into your home in the future will want to look at you and hear you talk, or watch you walk up to figure out, OK, that's

[INAUDIBLE]. I'll open the door for her before he gets here and maybe even say hello. I don't know.

Let me show you a video, though. It's kind of cute. Somebody using this technology in one possible scenario on this little smartphone-- this was actually a very futuristic smartphone back in the early 2000s that had a little camera on top. We could even have taken selfies back then if we knew what a selfie was, I suppose. And it can record you, and we'll see how you can use it.

[VIDEO PLAYBACK]

-Please say your name.

-TJ Hazen.

-Hello, TJ. Your identity is confirmed. Please say your name.

-Ken Steele.

-Please say the combo displayed on the screen.

-99-81-89.

-You are not Ken Steele. Please go away.

-Please say your name.

-TJ Hazen.

-Please say the combo displayed on the screen.

-25-94-46.

-You are not TJ Hazen. Please go away.

[END VIDEO PLAYBACK]

JIM GLASS: So this was a very simple movie, example, scenario showing how you can combine multi-modal inputs to do more powerful things. And I think these are important technologies that we'll need for our IoT devices in the future. And that brings me to the end of this multi-modality segment. Next, I'm going to talk about computational issues we should be thinking about in IoT.

HCI in an IoT World (Jim Glass): Video 6

Thus far, we've talked about the underlying technologies-- speech, understanding, different modalities. I want to shift gears a little bit in this segment and talk to you about some of the computational issues, or opportunities really, that we'll have in a world of IoT.

There's opportunities to do the computation either remotely on some kind of server or perhaps locally on the device. If you do it locally, though, you have to worry about power consumption. And this is where we're doing research on low-power language processing, speech processing in particular.

And where this could be interesting is if you have devices that are really just doing things locally, maybe like your Fitbit. You want to talk to your Fitbit about, I don't know, what you ate or what you did or something. Does it need to go off to the web to get information? Maybe not. Maybe you'll want to talk to something else. So there is a role, I think, for local computation.

And there are opportunities to do some things locally, some things remotely. It'll really depend on the scenario. And that's what I'm trying to convey on these lower pictures down here. In the top, we have things you might want to do in the cloud. So you could do voice activity detection, which is the problem of trying to figure out when somebody's actually speaking to the device. You're listening all the time. When are they speaking?

The Amazon Echo, for example, is constantly listening, but it really only wants to detect when somebody is addressing it, and then it will do further processing. It's an important technology and can help reduce power as opposed to continually streaming out the recording out over the web. You probably wouldn't want that to happen anyways for privacy reasons.

So you could do VAD in the local platform, and that probably makes a lot of sense. But then where you do the speech recognition? Do you do it out in the cloud, or you do it locally? Again, it depends on your scenario. And the same is true for understanding. You could do it locally. If you wanted to do everything locally, that could be possible. But it's also possible if you're getting information off the web somewhere, maybe you could do some of the processing remotely as well.

Let me talk a little bit about what we're doing to address the low-power issue for speech recognition and voice activity detection. This is a little bit of a complicated slide, but you can really break it down into two components. Speech recognition has a training phase where you have to learn all these models I talked about earlier-- the acoustic models, the language models, what's the vocabulary. And that can be done offline in software on powerful computers because they can take a lot of computation to train these models.

But then once you do that, you can compress them and you can download them onto a chip. And so if you have hardware, which is what we have down here, you can just have

speech recognition as a decoder. It's all trained. It's ready to go. You can talk to it. So you just need to do the search aspect of the speech recognition, and that you can do on a chip.

And you can combine it with other technologies like voice activity detection, which I talked about before. It's always listening. It's trying to figure when somebody's talking to the device. And then when it detects that, it has the audio go over into the recognizer where it can figure out what words are being spoken.

So we've actually been creating low-power circuits to do these types of things, and we're leveraging ultra low-power circuit design work. This is a collaboration, for example, I'm doing with Anantha Chandrakasan in the Electrical Engineering Computer Science Department, and we have chip implementations of a speech recognizer and voice activity detection.

Here's one instance of a chip. Now, it looks big on this slide, but it's actually only 2 and 1/2 millimeters, about a tenth of an inch. This is what it looks like when it's actually put in a circuit. The power this draws is actually 6 milliwatts, and we want to get much lower than that. And at that level, you could imagine putting them into devices.

We've also built a voice activity detector that just draws microwatts of power. So when you combine them together, these can get very small. And you can actually imagine putting these types of things in body-worn devices that one day could potentially be powered by our motion or temperature of the body. So it won't even need a battery. Imagine that. I think this is in the realm of possibility.

Right now we're working on the technology for a state-of-the-art speaker verification that I talked about in the multi-modality. So all of these technologies can potentially be implemented in low-power devices, which opens up many opportunities in the world of Internet of Things.

So that's all I'm going to say about computation. But it gives you the idea of, I think, the possibilities in the IT world. What I'm going to do next is move on to the next segment and talk about challenges for these technologies as we move forward.

HCI in an IoT World (Jim Glass): Video 7 Part 1

Now I want to move into the final area of this module where we talk about the challenges that are still remaining for the technology. And let's think about advanced multimodal interaction in the future.

We talked about why it's a really great thing to have. We want to develop the technology to let people interact with their devices, much like they interact with each other. And so what are the things we need to do that? It's a whole lot of different technologies when you think about it. And they all build on top of each other.

We need to do general activity detection. What on earth is going on? Who's there? Can we identify who they are? Can we verify their identity? Who's talking? If you have a group of people, which one's actually talking to the device? What are they saying? Maybe you want to listen to them, maybe you want to look at them to figure out. What are they doing? Are they doing other modalities that are important for the understanding? What's their intent? The understanding, do what I mean, not what I say type of thing.

And again, the paralinguistic aspect that I talked about earlier. How are they feeling? Are they happy, sad, or are they confused? These can all be important sources of information to enable natural interaction.

So I'm going to go through now and show you a bunch of different prototypes that address the challenges in different areas where the technology might be deployed in the future. The first prototype I want to show you is something that's in a car. And, of course, is technology has come out recently and is very popular. With things like the Ford SYNC, and now other manufacturers are following suit. And it's become very popular.

People are little unhappy with the technology. In fact, if you look at the complaints, people say the speech recognition is the number one thing they often complain about. So it needs to get better. And of course, in a noisy environment. It can be a challenging problem. With a microphone that are a little farther away from the person's mouth. Environmental noise can get in. It's a challenging thing.

So let's take a look.

Show Chinese restaurants near 32 Vassar Street in Cambridge.

There are 10 Chinese restaurants near 32 Vassar Street in Cambridge. The particular neighborhoods are Central Square, Inman Square, Kendall Square and Fenway.

Tell me about the New Mary Chung restaurant.

New Mary Chung restaurant is located at 460 Massachusetts Avenue in Central Square in Cambridge.

How late is Mary Chung open?

Here are the hours for New Mary Chung restaurant. From Wednesday to Thursday 11:30 AM to 10:00 PM.

One of the things that's interesting about interacting with the computer in a car is that-- especially if you're the driver-- even though you might have a nice display to show information, you don't look at it for safety reasons. So it actually reverts to an audio only communication channel. So think-- you need to understand that the computer needs to relay information verbally and be able to interact. That will be very important in the vehicle environment.

So let's take a look at another example with robots. I actually think robots are going to be the really cool device of the future. Eventually, they're going to be in our homes, they're going to be at work, they're going to be outside. And here's a classic example. Are you going to type on a keyboard to a robot? Of course not. You're going to want to talk to it.

This is the biggest robot I've ever had the opportunity to work with. It's an autonomous forklift, weighing several tons. And people would command it to go lift things up and put it on trucks, and things like that. So let's take a look.

This is actually a really crazy use of speech technology. You saw him talking to the forklift on a megaphone over 200 feet away. Now, there's a couple of interesting anecdotes I can tell you about that. First of all is, don't go away and let your graduate students play with the technology. Because they're the ones who discovered this. But it's actually amazing that it works so well from such a distance.

One of the things we were doing there you saw, we said, forklift come to receiving. That's the advantage, if you have a key word you're triggering on. So when people say, Siri, wake up. Or Alexa, do this. You can use that to help improve the recognition that, oh, they're actually talking to me. I should pay attention. And so you can do that.

This is not a scenario I imagine people using in the future. Although, you might want to call to your car, hey, come over and pick me up, in the future. Who knows? Maybe we can all be like Batman.

The other thing I want to point out about the robot that was interesting, is that this was originally designed as a safety feature. The forklift was about 15 feet tall. There was a lot of technology on the roof and it was very noisy. It's an outdoor environment. Gravel, uneven surfaces, there were engines. So we actually had array microphones all around the top of the forklift. It was continuously listening.

And it was supposed to be listening for people say stop if it was about to do something dangerous. There was a tablet that you could talk to and gesture and tell it to do stuff. The interesting thing is as soon as people realized that they didn't have to use the tablet, they

could just talk to the forklift, that's what they wanted to do. And that's an important lesson for the future in IoT.

We're not going to want to have little remote controls that we talked to our devices. Right now, for example, with Apple TV now has Siri, but you have to talk remote control. Ultimately, we just want to talk to the TV. You don't want to have all these little devices that you have to talk to the IoT devices themselves. You just want to talk to them directly. That's the way it should be, that's the way it eventually will be. Amazon Echo is a good example that.

The next thing I want to talk about is the kind of information we might need to get at. A lot of information, conventionally, has been stored in databases. A structured format, you can get it. The weather forecast, things like that. But a lot of information, as we all know, out on the web is very unstructured. Movie reviews, people telling you their opinions about restaurants, things like that. It's just very free form.

We can actually leverage that, summarize it, and give people that kind of information. But it's a challenge for language processing. Here's an example of an entertainment system we made a while ago that lets people speak in a very open ended way. Asking for particular kinds of movies. I'd really love to have this, myself, at home.

Show me a sci-fi movie about aliens from the last five years.

There are 41 science fiction movies about aliens since 2007.

Which was highly rated?

Eight of them were with the rating higher than seven.

I want to see an adventure movie about where werewolves and vampires.

There are two adventure movies about vampires and werewolves.

Show me the movie Pirates of the Caribbean.

There are four movies named Pirates of the Caribbean.

Show me the trailer of On Stranger Tides.

Any car chase movies about prisoners?

There are 26 movies about prisoners and car chase.

The important thing to remember about this video is they weren't just asking for titles or things like that. They're saying, werewolves, car chase scenes. You know, stuff that we don't know the title, exactly. We just sort of know, roughly, what it is we want to watch.

And you can actually get that information from user content. Things people talk about. So, leveraging that will make our interfaces much more powerful in the future.

HCI in an IoT World (Jim Glass): Video 7 Part 2

OK, I want to shift gears and talk about some of the challenge for contextual understanding that I mentioned before. Think about, for example, flights again, that prototype we showed at the very beginning of the module. Say somebody's saying, "Flights from Frankfurt to Boston." Well, you can look up some flights, show them.

But then what if people start playing around with that? It's very natural for us to do. And let me show you. What if I say, oh, "When does this one leave?" That's a form of verbal pointing, actually. And I might point to it at the same time. And the computer-- it's called Deixis-- the computer needs to figure out which one I'm referring to.

What if I then said, "What meal does it serve?" Well, what does it mean? OK, it's a pronoun. I need to figure out. You and I can do this, no problem. It's harder for the machine right now. Oh, "Show me the ones on United." Well, now I'm changing the game a little bit. So I have to inherit the fact that he is going from Frankfurt to Boston, but I have to go back several queries.

So this is what I mean by understanding the larger context and go, OK, United Flight from Frankfurt of Boston. Again, you and I, no problem. Computers, not so much. What if I then say "What about Lufthansa?" That's not a complete query. But people say, what about stuff all the time. It's a fragment. And we should be able to incorporate that information with the appropriate stuff that's been said before.

And, finally, "Give me flights to New York." Now I'm changing things, and I need to figure out-- well, I'm going to New York. Where am I going from? And actually this is confusing for humans. We might say, are you going to New York from Frankfurt? Or are you going from Boston? But we should figure out there's a problem and ask the right question.

These are all things that no devices out there right now can do. Our prototype could do it. It will eventually get it out there. But doing general purpose contextual understanding is a very hard problem. And it's a remaining challenge that it needs to be addressed.

Finally, everything I've been talking about is English. I'm talking to an English. All my demos are in English. But you know what? There are seven thousand languages spoken in the world. And ultimately, this type of technology should be available for everybody. So let's talk about multilinguality a little bit.

Now, one approach to dealing with multilingual interfaces is, again, remember I mentioned the meaning representation should be independent of the language? So you can have the same underlying technologies trained for different languages. Doing speech recognition, language understanding, speech synthesis, language generation. But once you get into the internal concepts and their relationship, it should be independent of the language. That's sort of an interlingua type approach. That's the term people use for it. And it could be useful for multilingual human computer interaction.

Let me show you one example of a crazy multilingual system where people are alternating speaking English and Japanese.

SPEAKER 1: What's the temperature in Boston?

[BEEP]

COMPUTER: In Boston this afternoon, high 70 to 75. Tonight low 60 to 65. Is there something else?

[BEEP]

SPEAKER 2: [SPEAKING JAPANESE]

[BEEP]

COMPUTER: [SPEAKING JAPANESE]

[BEEP]

SPEAKER 1: Could you repeat that?

[BEEP]

COMPUTER: In Kyoto, partly cloudy. High 27 Celsius. Low 18 Celsius. What else?

[BEEP]

SPEAKER 2: [SPEAKING JAPANESE]

[BEEP]

COMPUTER: [SPEAKING JAPANESE]

[BEEP]

SPEAKER 1: Will it rain?

[BEEP]

COMPUTER: In Tokyo, showers. Is there something else I can do for you?

SPEAKER 1: No, thank you. Good bye.

PROFESSOR: So in this movie you can see they're having a little dialogue, but they're changing the language. They're speaking in fragments. The computer is considering both

languages, figuring out which one's being spoken, and then answers in the language it thinks it recognized. A word like Tokyo could be spoken in either English or Japanese, so that's a little tricky. But it shows potential application of the technology.

And that brings me to the end of this module. I want to leave you with some final thoughts. Of course, I'm biased, but I think speech based interfaces for IOT is inevitable. Our devices are getting smaller. We want to talk to them all of the time. It's just so natural for people. We've crossed that point in our society where speech is out there and people want more of it. And I think that's what is going to happen.

These interfaces are the future. They have to be untethered. They have to be robust to different environments, different contexts. They have to understand in larger context. Have to incorporate different modalities. Have to be multilingual. The types of things we see out there now coming out of the commercial market on smartphones and other devices is just the tip of the iceberg. Much more remains to be done. There's lots of challenges, but the future is exciting.

Thank you so much for your attention. I hope this module has been useful for you. And I appreciate the opportunity to speak to you.

Robots and Autonomous Vehicles (John Leonard): Video 1

My name is John Leonard. I'm a professor at CSAIL studying robotics. I'm also associate department head in the Department of Mechanical Engineering, and today I want to talk about the Internet of Things as it applies to robots and sensing technology for robots. I'm going to talk about some sensors that we use in robotics, how we process them to acquire data, and a particular technology we call SLAM, Simultaneous Localization and Mapping, which is how a robot maps an unknown environment and uses that map to navigate. And then I'm going to talk about self-driving vehicles as an application of some of this technology that's generating tremendous excitement today. I'll talk about how a self-driving car works, what some of the open challenges are, and then I'll close with some thoughts on the future in what I think is a really promising field.

So I wanted to show some pictures of some of my favorite robots-- the MIT Urban Challenge Vehicle, some of our underwater robots, some robots that we've had wandering the corridors of MIT, some flying robots, and all of these robots have the problem of knowing their position in the world. Where am I? Where am I going? How should I get there? And the technology for how robots navigate through the world and build maps is critical also to the question of how we might take the Internet of Things into the physical world going beyond, say, RFID technology to using the ability to just recognize and locate objects in the environment.

So my vision for robots and the Internet of Things is what would it take to have the equivalent of search for the physical world? So when you ask Google a question, you enter a search query. It gives you an answer very rapidly. The way that that happens is that Google servers have crawled the internet to index and locate all the information on the internet. What would it take to do that for the physical world? How would you have the ability to keep track and know what is where in the world as a function of time?

And so my vision is that we build sort of an index database of the location of objects, people, entities in the world, and then we could track those over time, and be able to answer queries-- where are my car keys? Where is that book that I was reading? And the technology should be there to have this sort of physical internet, this sort web of things in the world that we can use robotic technology to answer queries about.

And so a big challenge though is to maintain this information as the world changes, and so it's the dynamics of the world that offers an opportunity to figure out information like to learn about objects and behaviors. It also presents a challenge, because the world is so dynamic. It's not a static thing. And so one way to think about it is imagine if all the objects in the world had RFID. It would make life easy as a roboticist, so shown here is a picture of a robot that's moving through the world.

When a robot travels through the world, if it obtains measurements to, say, RFID tags, those measurements provide sort of a link. It's kind of like a virtual link to a physical object, and as we collect multiple links to those objects moving through the world, it creates a graphical model. And so what I'd like to do is to go beyond, say, using RFID

and having the objects themselves tell us their identity, and then build up this sort of virtual internet, a virtual network of the observations of objects, where the robot can or a mobile device can automatically infer the locations of the objects and the location of the robot.

So in attempting this sort of generalization from RFID to dealing with objects in a natural state, there are some questions. And so the questions that come up are, what sensors are available? What technologies do we have for observing objects in the world? What is the right map representation? This has been just a critical question in robotics. How do we represent the world?

Going back decades, researchers have proposed many different types of map representation, and it's still an active research area. But one thing that we've sort of converged on is this. Graphical models are extremely powerful. And so the graphical models that embed location and observations of objects, to me, they form a sort of analogy to-- it's sort of an internet structure in some way for the actual world.

And then once we have the representation and the measurements, we need to perform inference. So we want to take all the measurements that we obtain and come up with an efficient and robust algorithm to combine that information to estimate locations for objects. And when we do this, we have to solve what we call the data association problem.

Data association is the task of identifying which measurements correspond to which objects in the world, and if we had perfect sensors, life would be a lot easier. But real measurements are uncertain, and we have matching errors that might be ambiguous. Is that chair one or chair two? Is that book A or book B? How do we handle the uncertainty in the correspondences between measurements and objects? And that makes for challenges to algorithm designers.

And then given these capabilities, as a roboticist, I want to build autonomous systems. I want to build safe self-driving cars. I want to build quad rotors that never crash. I want to build underwater vehicles that never get lost, that go off and find information in the ocean, and come back with valuable data. And so how do we create autonomous systems using these technologies? It's a big challenge.

And so my philosophy is that navigation, knowing your position, and building maps, knowing what's out there in the world, are big components of achieving autonomy. And then taking it further, my real dream is to build long-lived autonomous systems that can do lifelong learning, that can gather information over time and get better and better, and that can learn.

So a lifetime could be a matter of hours or days. It could even be months or years, say, for an underwater robot. And so for a car, imagine if a car could take the data from your daily commute, and every time you drive to work or drive home after work, there are new situations, new challenges, different weather conditions, different road patterns, but a lot

of the experience is the same. How could you enable a robot to learn over time almost in the same way that you would learn your commute after years like the back of your hand? And the robot would be on the lookout for the unexpected, for dealing with things that don't match the prior experience.

So my dream for sort of lifelong autonomy is a robot that gets better and better with time, and that actually is surprisingly hard, because sometimes it's easy to make a demo once, make a video, show a cool result, get it published in a paper, and so as a researcher, one of the biggest challenges I see is robustness, something that just works day in and day out, that you can trust like you trust that your car is going to start every morning. And those are still research challenges for us.

Robots and Autonomous Vehicles (John Leonard): Video 2

So now I want to talk about some of the sensors that we use in robotics for obtaining and representing location information. So, shown here are some examples of some of the robots I've used and sensors through the years. And on the left is a Polaroid sonar ranging system that was very popular in robotics say 20, 25 years ago. Extremely low cost, but because the device uses sound waves. So because the speed of sound is order of, say, 340 meters per second in air, it's pretty easy to make a circuit that can send out a sound pulse and measure the time it takes for the sound to reflect back.

In contrast, shown next is a laser scanner. What's called a 2D SICK laser scanner. That it has to time the speed of light. So, 3 times 10 to the 8th meters per second is how fast the pulses travel.

So, it took a little while longer for laser scanners to be used widely in robotics. And they're a lot more expensive. So the SICK laser scanner came onto the scene about 20 years ago at a cost of about \$5,000. But it really led to a huge surge in success in robot localization and mapping back in the late '90s, early 2000s.

But it still is at what we call a two dimensional laser scanner because it scans in a single plane and it returns about 180 measurements 75 times a second. That's great if your robot lives in a flat world with nice vertical walls, doorways, no steps to fall down. But for the three dimensional world, we need more.

And so the next picture is of Velodyne HDL-64 laser scanner and this was developed for the DARPA urban challenge back in 2006 and 2007. This is the sensor used by the Google car. So Velodyne gives you a million data points per second and it's a fully 3D picture. And so you can see-- as a human you can see buses and trees and people and the sidewalk and the data and its-- gives a much richer picture. It's still a challenge for a robot to interpret that data.

And the kind of things that a self driving car needs to do is to use that data to find cars and people, cyclists and try to predict what will happen next based on that data. Those are all active sensors. So, sonar and laser scanners they send out energy and then measure what comes back. But ourselves with our eyes, we use passive vision.

And shown next is a stereo camera, a Bumblebee stereo camera, widely used in robotics. And then just an example image from a single camera. And vision is our most powerful sense. We have just amazing capabilities for perceiving the world. And the progress in this field in the last few years has been tremendous, using technology such as deep learning.

But still vision is really hard for robots to develop, to achieve the sort of robustness that I talked about earlier. Vision can be fooled by difficult lighting and vision is just a really hard technology to work on. But the good news is that we're starting to see advances so that vision can be widely used in robot systems.

And then finally, I wanted to show the Microsoft Kinect, which is-- it's a camera. So RGB, red, green, blue. But it also gives depth, so we call it a RGB-D camera. And so many folks are familiar with this from the Kinect video gaming system. It uses a single camera combined with an infrared camera and a projected infrared light pattern that enables it to get depth information directly from the scene.

So won't work outdoors in direct sunlight, but for an indoor application, the Kinect can be really transformative because the ability to get color and depth information has enabled some really amazing new algorithms. In particular what we call dense mapping algorithms, that use GPU technology to build rich dense models of the scene in real time. Some of the techniques developed for the Kinect are now becoming available for stereo cameras that they might apply outside. So, I mentioned the map representation question earlier and I think we're at the point now where we can contemplate rich dense map representations for the world that are really exciting.

So, I wanted to go back in history to give a little context to show how lucky we are today with the sensors we have. And that is to show a little bit more about a Polaroid sonar system I mentioned. This is actually a video from my PhD thesis done at Oxford in the late 1980s. And here's a robot in a small room and some of the measurements are quite accurate and some of the measurements are just crazy. They're generated by specular reflections.

And so, one of the challenges was how do you operate with such noisy and uncertain sensors? And even though you wouldn't use sonar too much today, although the Tesla autopilot uses eight sonars that are pretty similar, quite amazingly. The principles that you see with using a noisy sensor help inform some of the way you need to think about getting more robust performance out of even better sensors. And so, some of the challenges include how do you model the physics of your sensor?

So, here you can model how sound waves reflect off of objects, how sound propagates in the air. We think about this a lot with underwater robots, how sound propagate underwater. So modeling sensor physics is important. Extracting features from the data. So, some of the data is going to be really accurate and reliable, say sonar reflections bouncing normally off the wall or bouncing in a corner, but other sonar measurements are going to be much less reliable.

And so if you can extract features, sort of extract the good data and then work on how you match those features for multiple vantage points. So, the video shows how combining data from multiple vantage points and doing what we call outlier rejection, throwing out the bad data and keeping the good data, and solving for the data association, as I mentioned, which measurements correspond to which features, then you can combine the data to estimate the geometry of the world and the motion of sensor.

Next, I wanted to show some LiDAR data. This is data from the HTL-64 sensor from Velodyne and this is the sensor on the Google car, as I mentioned. This is a video of taking the Velodyne sensor and driving it down Massachusetts Avenue from MIT to

Harvard. And what you can see in the video are buses and people and trees and buildings and you see the richness of the LiDAR. If you compare the LiDAR with the sonar, you can see that now you have a much better idea as a person, what's going on in the world. And you have this real rich source of information.

The sort of processing we need to do on a LiDAR system is to cluster measurements together to extract sort of what might be objects and then to try to classify them. And Google are the best in the world at doing this. Right now, they have amazing capabilities for tracking cyclists, pedestrians, and other vehicles using pretty much this type of data. They have developed their own LiDARs, which is pretty amazing, in just the last few years.

And so LiDAR is a wonderful sensor, but there is a catch. The catch is Velodyne LiDAR costs \$75,000, so it's a very expensive sensor. So, a dream would be to achieve the capabilities of LiDAR but with vision because cameras are so ubiquitous and cheap.

So next, I wanted to say a little bit about computer vision. The capabilities of vision are just amazing. It would be a dream to have a robot that could see as well as a person or even better. But the challenges are still pretty great.

I wanted to show a couple videos of some recent research results from my research group-- one of my PhD students. The first video shows stereo vision processing of a driving data set from Germany that was released open source on the internet. And the video shows a high speed algorithm that computes 3D point clouds, a bit like the Velodyne laser scanner data, but just from passive vision. And so the good news is that this is much cheaper than the Velodyne and the richness of the data is quite compelling.

On the other hand, you do see things like shadows and you can see some of the challenges that a robot would face using vision to interpret scenes in the world. The next video shows results from processing data from-- it's actually a Kinect data set. So as I mentioned, an RGB-D camera. This was actually a public data set from the University of Washington. And this is one of my students'-- the same student, Sadeep who's done an amazing job, processing the data to try to combine object recognition with what we call SLAM, which I'll say a lot more about later in this lecture.

And the video shows using an object detection system that is location aware. It uses a knowledge of the location, the motion of the camera, and building online a reconstruction of the scene, to do much better performance labeling objects. So those are just two examples of computer vision algorithms. And just to sort of summarize, the ability to build 3D models of the world in real time with stereo processing shown on the left and the ability to recognize objects while concurrently mapping their locations is shown on the right.

Next, I wanted to show an example of processing the data from a Kinect RGB-D camera that is carried through the world. So, there's not a robot here, it's just a person carrying a

mobile device. And this example shows how with modern GPU technology you can build rich, dense models of the world in real time.

This particular algorithm is called continuous. It's a spatially extended version of the KinectFusion algorithm. And it shows how if you take SLAM-like concepts of representing uncertainty and combining data from multiple vantage points, you can just walk through the world and in real time, build a rich 3D model of the scene.

Here's an example of when the video is finished, the sort of rich models that you can build. And you can see the quality of the reconstruction in the bathroom. The tub, the sink, the level of 3D detail. As a roboticist who 25 years ago, just getting down a corridor with the sort of limited sensors we had the time was a marvel, and now, we can really experience and capture the rich 3D world.

Next, I want to mention one more important source of information and that's inertial sensing. Inertial sensors use accelerometers and gyroscopes to integrate a position estimate over time. They suffer from drift that accumulates, but they're incredibly valuable and useful. And we have-- now we have these devices in our cell phones and they're becoming ubiquitous.

In this particular example, I want to show the combination of visual information with inertial information. And this is an example from MIT research, in which the goal is to combine what we call dead-reckoning. So integrating the motion of a sensor over time from the inertial sensors with visual information about the world to estimate the motion of a robot or a mobile device.

In this example, it's a camera, a single monocular camera and MEMS inertial sensor mounted together and then just carried through the world. And typically, these algorithms operate using a common filter algorithm. And there's some great research about how to develop the right sort of common filters that maintain consistency while still being efficient and giving good answers.

In the left in the video, we can see features that are extracted from the image. And so these features are tracked over time. And they're combined with the inertial information to compute an estimate of the trajectory of the mobile device, which is shown on the right.

This sort of capability of combining information about the world with what we called proprioceptive sensor measurements, information about ourselves, about how a mobile device or a robot moves, these are tremendously powerful. And my prediction is these will become ubiquitous in mobile devices over the coming years. And here we can see an example for a different experiment with that visual inertial device where the user walked up a set of stairs and then back down again. And you can see the comparison between the estimate of the algorithm, which tracks of visual information. And you can see that it does a really good job of tracking the motion of the person up and down the stairs.

OK, now I want to mention one more type of technology which I think in an internet of things context which is a way of connecting sort of the RFID world where we have tags on objects with the robotic perception where we don't. And that's to use visual fiducials to help us develop some location based algorithms. And so, in our education and research at MIT, I'm quite enamored with something called AprilTags, which were developed at the University of Michigan by one of our former students, Ed Olson, he's a professor there. And these are visual barcodes that provide a unique ID and were developed for robotic applications.

And so there's a whole different family of codes providing different numbers of tags in different levels of robustness. They're really, really useful. And so they're fully open source and they enable efficient testing of algorithms for estimating locations. And some of the ways that we use them are to put them on robots. So, shown here are some robots, TurtleBots, that have AprilTag cubes on top of them.

These robots are operating in a motion captured system. You can see the motion capture cameras mounted up in the ceiling. My dream is to make motion capture systems unnecessary, that the robots can just sense the natural world and estimate their location as accurately as they would by motion capture system. Which is a pretty lofty goal because they're very, very accurate.

If you're a student, my advice to get your feet wet in robotic perception and mapping research, is to look at AprilTags and to use AprilTags as an easy way to get exposed to some of the location estimation algorithms, while not having to solve the data association problem. So, here's one more example of using AprilTags. This is from a paper we submitted for publication last year. And it shows the original images, AprilTag fiducials extracted from the images, the way that they're used in what's called a bundle adjustment algorithm to estimate the scene geometry and the motion of a camera, and then re-projecting the projected tag points back into the original image. This gives an illustration of how accurate the AprilTags are and how easy they are to use for testing your algorithms.

Robots and Autonomous Vehicles (John Leonard): Video 3 Part 1

Now I want to talk about simultaneous localization and mapping, or SLAM. Which is the problem of how a robot builds a map of an unknown environment and uses that map to estimate its position.

I've worked on this problem of SLAM my whole career, and it's been amazing to see how the field has evolved over the last 25 years. We now have quite highly capable systems that are being deployed.

And so for example, SLAM is at the heart of how the Google self-driving car works. Before the car can be driven autonomously, in advance, a human operator drives the car around the world collecting data that gets combined with SLAM algorithms to make rich, detailed maps. And then those maps are used to localize the car very accurately.

So what I want to do here is to just try to give the essence of SLAM with a few examples. Talk about some of the issues and some of the approaches to attacking SLAM.

So now I want to play a video for SLAM that sort of a famous video. It's what's called the Victoria Park data set from Sydney, Australia. This is a public data set that was published back in around 2002. And this is a video from Michael Kaess, from Georgia Tech and MIT that's public. That shows the processing for a pickup truck driving around a park in Australia, making measurements of trees using a SICK laser scanner. And features are extracted from the laser data to give range measurements and bearing measurements to the trees. And the goal is to combine the data to estimate the trajectory of the pickup truck and the location to the trees.

And so the video shows the four different components of a SLAM solution. You see the original trajectory of the car is computed by dead reckoning. So we have the dead reckoning trajectory shown in blue. We have the estimated landmark locations, shown in green. The measurements are shown in orange. And then what the SLAM algorithm produces is shown in red.

And what you can see is the SLAM algorithm estimates the trajectory of the vehicle going back to where it started, doing what we call loop closing. So, loop closing is when a robot starts at one place and goes back to that place via different route. And it's able to compute that it's back where it started.

And so you can see the essence of the problem, that the blue dead reckoning measurements drift. The error grows without bound. And what we do with SLAM is we bound the error. We get an estimate that takes away that drift and produces a consistent estimate of both the trajectory of the robot and the map of the world.

I wanted to dive in a little deeper on this data set and show you the raw sensor data from the view of the robot. So this video now shows if you were in the reference frame moving with the robot, what would you see?

What you can see are the laser scanner measurements. And you see clusters of points that correspond to each feature in the map, in this case a tree trunk. And one algorithm groups those together to create a sort of composite measurement of the range and bearing to the tree. And then the SLAM algorithm takes those measurements and fuses it all together to get the map and the estimate of the trajectory of the robot.

So the essence of SLAM is to take those relative measurements shown in the previous video and then combine them together. And shown in this video, you see the processing, again, of how those are combined together to estimate the robot's trajectory. And also map the feature locations.

So, now I want to say why is SLAM difficult? Why has this been such an important and rich research area? And I like to think about problems like this along different axes of difficulty. On the right hand axis, we have this representational question. How do we represent the world? Do we use rigid global reference frames? Or relative reference frames that are more sort of rubbery and stretchy? Do we use metric versus topological representations? Do we represent the world in terms of objects? There are a lot of great questions about representation.

Once we've chosen representation, we then have this challenging inference problem, which mixes typical continuous estimation problem. So, things like common filtering location estimation with discrete decision making about the correspondence of measurements. Solving for data association and answering this loop closing question. Am I back where I started?

And then the other axis of difficulty is that we want to build systems that are robust and autonomous. And so how do you close the loop of taking the inference algorithm and the representation and use that actually control a robot? So guiding the motion of a robot so that it's safe, so that gets information about the world.

So thinking about SLAM, there are two main approaches. One is a filtering approach, which really comes from common filtering, which is widespread and ubiquitous in the position estimation community. And in filtering approaches to SLAM, we use a state vector with gaussian measurements of error that represents the location of the robot and the location of the features in the world. And associated with that is a covariance matrix that captures the uncertainty.

And the common filtering approaches were attractive in the early years of robotics research because of their computational efficiency. We had such limited computation available that it was almost unthinkable to do anything else. Because just getting a filtering approach to run in real time was a challenge in itself.

In the past 10 years another approaches emerged, and we call that pose graph optimization. And in pose graph optimization approaches we represent the complete robot trajectory. And we solve a nonlinearly squares optimization problem to estimate the full trajectory of the robot and the features, or the entities, in the map.

So here I want to show two examples. The first is an old example from test we did at MIT in an underwater tank using sonar. Of having a robot navigate around, making measurements of point features. This is a bit like the Victoria Park data set. But this video shows the visualization of the error covariances for the features.

And a key thing to get from this video is that, initially the error covariances are large. But as the robot moves around and get some more information, the error covariances shrink. And in filtering approach, a key thing is to maintain consistency so that you don't reuse information twice. You properly account for the correlations between the estimates for the different features. And if you don't do it right, what will happen is the error of covariances will keep shrinking way too small then what they really should be in reality. And so one of the challenges is achieving a consistent state estimator.

So even today there's some great work-- for example, with the Google Tango-- of applying the latest algorithms and consistent filtering to a mobile device. So that it can operate for a very long time, giving reliable error estimates the entire way.

The second video I want to show is SLAM using vision. And this is now fully three dimensional, and uses pose graph optimization. So this is work from my Ph.D. student Harder Johansson using data from a PR2 robot navigating around the second floor of this data center at MIT.

And shown here, you can see the back projected data from the estimated trajectory locations of the robot. The robot's building a fully three dimensional map. And you can see the loop closing occurring as the robot travels back and forth down a corridor. You see the little corrections that happen as the robot's traveling through the world. And so at the end of the day, we get an estimate of the trajectory the robot. And an estimate of a map of the world.

So I wanted to show one more filtering example. This is from something we did at MIT about 12 years ago. Back when we had 2D SICK laser scanners. This is on a B21 robot. And if you think about The Internet of Things and RFID and this notion that objects in the world have tags. Wouldn't it be great if everything in the world was a landmark that said, this is who I am.

And for testing algorithms back before we had fully capable 3D computer vision, we did a test where we used the hurdles of the MIT track team on the indoor track. We set them up on the indoor tennis courts at regular locations around the court so that we could easily measure the ground truth locations. And then we steer the B21 robot with a 2D SICK laser scanner around the tennis courts.

And what you can see here is an estimate of the trajectory of the robot from dead reckoning, shown in blue. The sort of squiggly pattern that just goes crazy all over the place. And that shows you how quickly a robot can get lost just using wheeling coder data. We call that odometry. And for this kind of robot, which had four wheels, changing

which wheels are in contact with the ground, the varying levels-- it quickly gets really lost.

And the beauty of the SLAM algorithm is that with no prior information the robot can get a really accurate estimates of its trajectory, and also map all the locations of the features. And here we can see a comparison against ground truth. So what we measured by hand, versus what the robot measured. And they agree perfectly.

Robots and Autonomous Vehicles (John Leonard): Video 3 Part 2

Next I want to give a 2D example of pose graph optimization, and this is a famous data set we took at MIT called the Killian Court dataset. Back in 2002, we navigated a B21 robot by hand around MIT's Infinite Corridor and some of the related buildings, and this provides a really good example of the loop closing problem.

This processing is actually from 2006. Our student, Ed Olsen, used a pose graph optimization algorithm where we represent the world with all the positions of the robots, so shown here is the triangles. The robot positions get saved, and then as the dead reckoning error accrues, you can see that the parts of the map don't align to one another. And the really cool thing is when you apply the loop closing constraints, which happens at the end of the video, you get the correction. So you can see the before and after. And one of the great advances in the field was being able to do these corrections very quickly. So with the right map representation, thinking about the sparsity of the underlying map representation, it's possible to apply the constraints for loop closing very quickly and get a corrected map.

So since pose graph optimization is the dominant paradigm in SLAM, I wanted to go through an example in a little more detail of how it works. And so shown here, I have a PR2 robot. It starts off with no knowledge of the world. So at time equals 0, it doesn't have any prior information. And we want to use its onboard sensors, which might be wheel encoders, inertial sensors, and the sensors that it has onboard for sensing the environment-- laser scanners, cameras, perhaps sonars. And we want to combine that data as the robot moves through the world, building up this virtual network representation of the world.

So initially the robot starts, and we'll just arbitrarily say that'll be the origin of our coordinate system, so where the robot starts. Before it moves, let's say the robot takes a measurement of an object. Let's say it's a chair. This is a landmark, and so m_1 designates the measurement of that first object. It might be a range measurement or a range and bearing measurement.

And then the next is the robot's going to move. So at $t = 1$, the robot moves to its new location. That's based on the control input we call u_1 . So u are the control inputs and m are the measurements. And the control inputs, we typically call that odometry, typically derived from wheel encoders, say, for a wheeled robot like this robot.

Now, before the robot moves again, it's going to make an observation of that first object, the chair. We'll call that m_2 , measurement 2, and then it's going to measure a new object, m_3 -- say it's a table. And the robot's going to move to the world taking measurements of objects. All the measurements are relative. Its motion estimates are relative and its measurements of objects are relative. What we want to do is combine all that relative information efficiently together to create a global map representation.

And so if we say the robot's going through the world for some amount of time, and now we're up at time $t = n - 1$, we get another measurement of the table-- we'll call that m_4 . And then the robot's going to move one more time-- $t = n$ -- and what we want to do is take all that data and put it in an efficient representation that we can use for estimation.

And we call this a factor graph representation. And so factor graphs are well-known graphical models that provide a nice clear way to represent the conditional independence relationships between estimates in a graphical model problem. And I want to give credit to my colleague Frank Dellaert at Georgia Tech as one of the champions of the factor graph representation, which has been really effective and powerful for SLAM.

And so here's an illustration of a factor graph where we've taken that earlier example where we had a chair and a table as the landmarks, and now we're going to abstract that and we're going to have L_1 and L_2 as landmark positions. I mention those control inputs u_1, u_2 , and so forth, and there's one other type of constraint. Each measurement is a constraint, and the new type we have to add is the loop closing constraints, which we saw in earlier videos when the robot went around the Infinite Corridor and came back, and there was a big error, but we could correct it. And so these are non-sequential constraints, and these are shown as c_1 and c_2 .

And so the factor graph is a bipartite graph which has variable nodes and factor nodes. And effectively we want to estimate the variables given the information conveyed by the factors. And there are algorithms that do this. So to fully go into the factor graph math, it's probably more time than I have now, but I just want to quickly give the essence of the problem and how it gets approached.

And so what we do is we define a cost function. So what we have here is a big inference problem. So remember that the robot might be operating a very long time. There could be days, weeks, months of data, and there could be hundreds, thousands, millions of features-- say, a robot taxi going around Manhattan continuously for days, weeks, or months. Now, you probably wouldn't attempt SLAM at such a space and time scale, but the maps can be very big, and it's a very large-scale inference problem.

And so to try to succinctly convey what the essence of some of the challenges here are, it's really about maintaining the conditional independence of measurements. So because all the measurements are relative, if we do things in a clever way, we can prevent estimation in one part of the map propagating to another part of the map unless it's really necessary due to a loop closing. So just a little bit of math to try to explain the essence of the problem.

The cool thing is that we can take this $f(\theta)$, the big thing that we want to estimate, and break it into a product of factors. And so this is a product of factors, and then we basically can take the different pieces and break them out individually, and we can attack different subsets of the problem. A big thing is that as the robot moves, the map is getting built up over time.

So, say, when the robot goes from this time step to that time step, here we're adding in this new piece, and now we're going to add in the loop closure constraint here. The good news is that because of the conditional independence relationships, this sort of part, we can take this part and prevent it from changing what we had to do with this part of the map. So just very briefly, the essence of it is to pose it as a maximum likelihood estimation problem, so we're going to try to find the parameters for theta that maximize this function f . And the way that we do it typically is we assume Gaussian errors, and so when we assume Gaussian errors, we can represent them as normal distributions with a mean and a covariance, and then we can take this and use the sort of log trick to make this as minimizing the negative log of the function f , and representing that as the summation over all the different factors.

To do this in practice today, there are some great software packages out there, there are some great algorithms that are out there. And places I would recommend people to look definitely look for GTSAM from the Dellaert group at Georgia Tech, g2o from the Burgard group at Freiberg, and then ISAM is from MIT, from Michael Case, and formally in my own group.

So now I wanted to go back to that example I showed before, the visual SLAM on the second floor this data center. And so I'm replaying the video, showing the processing of the robot, and maybe you can look for the loop closures and look for the sort of corrections that happen. But I'm also showing on the slide here the trajectory of the robot. So at the end of operating all around the second floor for several hours, you see an estimated robot trajectory, and then you see the map that's created by back projecting sensor measurements from the trajectory locations that are estimated and then comparing that against the floor plan of the building.

So really the essence of pose graph optimization is to say the trajectory of the robot is just an essential part of the representation. And once we have the trajectory of the robot, we can then back project measurements to get a map. So this has been one of the great advances of the last 10 years in the field because we've been able to achieve quite successful results. But there is still the catch that representing the entire trajectory of the robot isn't going to be scalable with time. So at a certain point, something has to change because if the trajectory just grows and grows without bound, that's not going to work in the long term.

Robots and Autonomous Vehicles (John Leonard): Video 4

So I wanted to give one more example of Pose Graph optimization with iSAM in this case using text as landmarks. So in the vision I have for the sort of Google for the physical world of knowing about objects, and people and things in the environment as a sort of network, as a sort of virtual internet, text is going to be a really important information. So imagine if all the text in the world, all the signs, exit signs, directional signs in train stations and shopping malls-- if they were all sort of captured into a map representation that then could be used for users with mobile devices.

So in this example, we're going to show using text as landmarks. And for this test, we used AprilTags combined with some text that we printed out as landmarks to show how the SLAM algorithm works. The motivation for this application was actually partly for having visually impaired folks navigate using mobile devices. So imagine if mobile devices could read text in the environment and then communicate that to a visually impaired user so that they could navigate, say through a train station or a shopping mall.

So shown here on the picture is some text, in this case, the word score with an AprilTag beneath it. And then, below that we see the text detection coming from an OCR, Optical Character Recognition, algorithm, and correlating that to the AprilTag so that we can calibrate it, and we can ground truth that we're getting the right detections. For doing SLAM with text, we have the same graphical model representation as shown before. And in this case, the camera poses are the states, the trajectory.

Again, we used the graphical model, a factor graph, and instead of a robot trajectory, we now have the camera poses for our mobile device. And the landmarks in the environment are the locations of text that occurs on signs. So for this example, we used the Google Tango device as the input sensor, and we walked through the world, and we placed on the walls most of the words from Lincoln's Gettysburg Address. So "four score and seven years ago our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal."

So we took all the longer words, and we put them onto the walls with AprilTags. And then we walked around repeatedly to show how we might automatically capture text and then use that to navigate. So here I want to show a video of Pose Graph SLAM with text landmarks. And you can see the optical character recognition algorithm working on the text, and you can also see the AprilTags being detected. And as the mobile devices move through the world, these detections are added as information, as constraints into the factor graph representation.

And then using the iSAM code developed by Michael Case, the constraints get sort of fused together to come up with an estimate of the location of the text signs in the world, and also the trajectory of the device. In the graph, we show the dead reckoned trajectory from odometry, shown with time as the vertical axis. And this shows the multiple loops through the world, and how that drifts with time. So using what we call text spotting

information, we use text as landmarks to constrain the motion error for a robot or a person going through the world.

So here we can see the output of the SLAM algorithm. And you can see on the right, you see the estimated tag locations, and then the time is the vertical axis. And so, as the user goes around, and around, and around, you see the sort of trajectory that gets estimated. And so, here in one of the input images you can see the text that's detected, and also the AprilTag, which we use for ground truthing. And then, down here you can see that without SLAM, which just the drift from the free inertia of the system, the tag locations are scattered. They don't line up with one another.

But then, using SLAM, we get consistent estimates for the tag locations. And so, using the factor graph representation combined with iSAM, we can fuse together all the information to get the locations of all the signs, and the trajectory followed by the device through the world. So next, I wanted to show Pose Graph SLAM with loop closing for the dense data that I mentioned earlier.

So I mentioned how we had a spatially-extended version of the KinectFusion algorithm that can enable a mobile device, a Kinect, to be moved to the world to build a rich, detailed model. Now, for the example I showed earlier of the small apartment scene, the consistency was pretty good. But if you go over a large enough area, you're going to get drift. And so, this example here shows eight minutes of walking on two different floors of this data-center, up and down stairs, and showing the drift that occurs, and then the correction that gets applied when you combine iSAM and Pose Graph estimation with deformation-based loop closure.

So now we're taking techniques from computer graphics to deform the mesh as created to match one another when we have a loop closure. And so, for example, in the video you could see at the beginning there were two blue chairs that were misaligned. And when we come back and apply the correction to line up the two blue chairs, the rest of the map gets corrected automatically. And then, you can see a rich, beautiful model that we can fly through.

And so, here you can see the ability, the potential, to grab rich 3D models, including loop closure-- so correcting drift and building models of the world that really should be pretty useful for a lot of applications. And building on that theme of the rich, dense models with loop closure, I wanted to choose an example from the MIT DARPA Robotics Challenge Team. So this is not my research, but my collaborators did some really cool work for the Atlas robot, which was used in the June, 2015 DARPA Robotics Challenge.

This is a robot, a humanoid robot, doing a disaster response task. This is from a test done at MIT where, using stereo vision input to the KinectFusion algorithm, the team were able to estimate in real-time the locations of these cinder blocks, steps, and then control the robot to autonomously choose the footstep placements to walk over that terrain, and traverse up and down the stairs. And to me, this is one of the coolest things I've ever seen, and I've been in robotics for 25 years.

But this combines humanoid motion planning and control with dense 3D position estimation and SLAM. And it shows the potential that if you could really combine perception robustly with motion planning and control, you can enable robots to do really challenging and useful tasks. So for this work, I want to give great credit to Russ Tedrake, the leader of the MIT DARPA Robotics Challenge Team and his staff, including Scott Kuindersma who's now at Harvard, and Maurice Fallon who's now at the University of Edinburgh.

Robots and Autonomous Vehicles (John Leonard): Video 5

Now, I'm going to switch gears and talk about self-driving vehicles as an application of some of this technology. And so this photo shows the DARPA Urban Challenge vehicle from 2007. I was the team leader with an amazing team of MIT faculty, post-docs, and students. And this was some last minute late night hacking with Ed Olson, who developed AprilTags in the post-graph SLAM optimization algorithm with me, in the backseat of our robot car, and Luke Fletcher, one of our post-docs, in the front seat. Self-driving vehicles are just such an amazing technology. You just have to look at the newspaper today to see all the excitement and all the activity, and so much of this is due to the Google Self-driving car project and all the other efforts that have followed it.

And so for self-driving vehicles, I have mixed motions because I see SLAM. I see all the technology I've talked about with sensors, and uncertainty, and inference, and map representation, and autonomy all coming together for a compelling societal problem. On the other hand, there are still some great challenges. So what I want to try to do is talk a bit about how self-driving vehicles work, and then how the SLAM technology feeds into it, but then, also, some more general questions about open challenges for perception and autonomous decision-making that we'll need to solve before self-driving vehicles can become widespread.

So there's incredible excitement in the field due to the Google project and all the other activities that have happened, folks like Tesla, and Uber, and the major car companies. And so on the one hand, we have this tremendous excitement. But on the other hand, we have a lot of questions. And there are still technology questions and how self-driving vehicles might impact energy use and the environment.

So there isn't time to address all these questions, but I want to try to give a little taste of some of the technology and some of the questions that I think about. So how does self-driving vehicles connect to what I've talked about so far? Well, location information is absolutely critical for self-driving vehicles.

And so a self-driving vehicle needs to answer, where am I? Where am I going? How should I get there? What is around me? What will happen next? And then, finally, what should I do next? There's no doubt that location information is absolutely vital for self-driving car technology.

First, I want to talk about some of the potential benefits. And so the safety benefits alone, to me, just make it so important to work on this problem. Sadly, there are 5 million vehicle crashes per year in the US. 93% of accidents have human error as a primary factor. And there are over 30,000 fatalities per year in the US from traffic accidents, and worldwide, it's over a million. And that's really sad.

Now beyond the safety applications, we have the potential to increase the efficiency of our road network, and also, people could recover time lost due to commuting. If you didn't have to pay attention to the road, and you were safe, you could work on your

laptop. You could watch movies. You could do lots of things. Some people say you could even go to sleep-- I think that's still going to be a long time away when you can sleep in your car.

And if you think really ambitiously, if this technology came to fruition, I can imagine radically new models of the way in which goods and services and people get distributed, especially in cities. So you can imagine urban parking lot space being reclaimed for places for people to live. You could imagine vehicles that just appear on demand when you need to get from A to B. And think about a Mobility on Demand system with driverless vehicles. There's the potential that you could really change the way cities are structured.

And sometimes I think about having mobile Starbucks vans, and I think about them in a park. And wouldn't it be nice to have a latte and have a drone fly from this mobile Starbucks van with a latte and fly it to me sitting on a park bench. That maybe is a little crazy. But I think there are new ways to think about how things and people move through the world, and self-driving vehicles might enable that.

So next I want to say a little bit about how self-driving vehicles work. And I'm going to use the example of our DARPA Urban Challenge Vehicle from 2007. So this vehicle is called Talos. It was a Land Rover LR3, and it was built with an amazing team of colleagues. And we had quite a computer system on our vehicle.

So we had a blade cluster, which at the time had 10 blades, each with four cores. So we have 40 cores. We didn't want to be limited by computation, so we wanted to have a lot of computational power to do advanced vision algorithms. We had a lot of sensors. We had an Applanix Inertial GPS sensors.

We had 12 of the SICK 2D laser scanners. We had a Velodyne, which I talked about before, on the roof with its million data points a second. We had 15 radars looking out the front of the car at various angles. And we had five cameras.

To power the computer system and all this, we needed a six kilowatt generator in the back. We had a two kilowatt air conditioner on the roof just to cool the blade cluster. Fully loaded, the blade cluster took 3.5 kilowatts of power, which is a little crazy, but we didn't want to be limited by computation. And we wanted to have as many sensors as we could fit on the vehicle.

Here, I want to show a video of our car driving. The 2007 Urban Challenge was one of the most exhausting and rewarding research experiences of my life. We had a very short amount of time, a year and a half, to develop a fully autonomous self-driving vehicle. And we had to deploy it in a race that was about 60 miles driving autonomously in traffic.

And the race took place in Victorville, California. There were 11 teams that made it to the finals, and six teams finished the race. Carnegie Mellon came in first. Stanford came in

second. We came in a pretty distant fourth place, but we were proud of the research that we put in our vehicle, advanced algorithms for motion planning and perception.

So next I want to show how the car actually works. We're going to play video that shows the internal map representation of the car as it navigates through the world. And so the robot builds a local map using its perceptual data. It's in a local reference frame that moves with the vehicle through the world.

And you can see the motion planner system generates random potential paths, 50 paths, 10 times a second. And it's continually trying to choose what is the best path. And the way that it makes a decision about which is the best path is by rendering that path into the local model, which has things like obstacles and other vehicles, and tries to decide which are the safest paths, which are the best paths.

And our system architecture shows the different components of the system. On the right hand side, we have our sensor data feeding into perception algorithms. These try to find where is the road? Where are the obstacles? Where are the curves?

And they populate that local map representation that I mentioned. The beneath perception is something called the Vehicle State Estimator that processes the inertial data and the wheel odometry from the car to estimated vehicle trajectory. And this local map-- we call it the grid map-- that provides the conductivity to the motion planner that I mentioned.

We used a rapidly exploring random tree, RRT motion planner. And at the time it was one of most sophisticated algorithms available. And it's a really powerful algorithm, but it uses a lot of computation.

And since then, actually, partly as an outcome of this project, we've made advances to that algorithm. So my colleagues Surtash Karaman created the RRT star algorithm, which is widely accepted today as the sort of right way to do this, to have a randomized motion planner that generates good paths. And so the motion planner takes commands from a high-level navigator that tries to reason about how to get from one point of the world to another, navigating through the map given by the competition organizers, while also obeying the rules of the road, so stopping at stop signs, staying in the right lanes, and so forth.

And that feeds down to a low-level Vehicle Controller. We used a pure pursuit controller, developed by my colleague, John [? Hower ?] and his students, that accurately controlled the vehicle along the commanded paths. Overall, to realize the system architecture, the team had to create about 150,000 lines of new software code.

Much of that, we've now made available open source. So our data logs on the race are available on the internet. And we feel that when you approach a project like this, the more tools you can give to the community via open source, then that gives the greatest impact for long-term research.

Robots and Autonomous Vehicles (John Leonard): Video 6

So the DARPA Urban Challenge was really important in the history of self-driving vehicles, because it led directly into the Google Self-driving Car project. Back in 2007, Carnegie Mellon came in first and Stanford came in second, and it turned out that the leaders-- the technical leaders of those teams were secretly hired by Google X to start the Google Self-driving Car project. And so, during sort of the 2009, 2010 time frame, key members of the team joined Sebastian Thrun to try to create the self-driving vehicle, and they made amazing progress. So they performed 10 100 mile routes autonomously various parts of the Bay Area. And total, they drove 140,000 miles autonomously while the project was still totally in stealth mode.

They were sort of ousted by John Markov of the New York Times in October 2010, and that was kind of a watershed moment in mobile robotics because it was like, wow, this is really happening. And so since then, they had continued to work with their Prius vehicles. They did an amazing video of taking Steve Mann, a blind person on a trip to Taco Bell, and that generated about six million hits on YouTube, and suddenly, the self-driving car was here.

After that, they switched to the Lexus vehicles, and I got a chance to ride in one in 2014 with my PhD student Russ [? Finnman, ?] and it was amazing. They had developed a highway auto pilot project, something called the Google Chauffeur, they called it. This is similar to what would be called a level three vehicle, and I'll get into that a bit more later, about how you have a system that does most of the driving. It's partially autonomous. It might do 99% of the driving, but a human has to pay attention for that 1% to take over control.

And Google decided, after their pilot of the highway driving, to shift to what's called level four driving, fully autonomous driving, on city streets. And so they shifted to city streets with the Lexus vehicle, and then they built their own small prototype vehicles that don't even have a steering wheel, or break pedals, or an accelerator. And so, if you look back to 2007, now today, eight years later, it's amazing what happened. And I think a lot of credit really goes to the Google team-- Chris Urmson, Sebastian Thrun, and all their colleagues for just believing it was possible.

As much as I'm a huge fan of the Google project, I do want to talk about some of the technological challenges. And, really, I see four key challenges for self-driving vehicles today: maintaining the maps that the vehicle uses, dealing with adverse weather, interacting with people, and achieving robust computer vision-- getting to near perfect detection with very low false alarms.

And so, I get asked a lot of times, when will we see the self-driving car? And I try to convey to folks that, yes, there's been amazing progress, but there are still these research issues and things like SLAM that are still, in my opinion, fundamental research. And so, how do you convey what the challenges are while still keeping that promise and that sense of excitement?

So what I did was, I put a dash cam on my car, and I captured about a month of my commuting in Boston back and forth from where I live in Newton mass to MIT. And it didn't take me long in Boston to find tricky driving situations. What I want to do is play a few videos that I use to show examples of technological challenges for self-driving vehicle technology. And some of these are very challenging issues in human computer interaction, computer vision, some of them relate to SLAM and mapping, and localization, and some relate to developing better sensors.

The first challenge that I mentioned is maintaining maps. So the Google project relies on a very accurate map of the world that measures the reflectivity of the laser scanner off of the road surface to create a sort of appearance map combined with geometry of the local road scene, and that's used to estimate very precisely the location of the car. Remember I said, where I am, is the first question. The Google car answers where am I down to centimeter level accuracy, tens of centimeters or better-- a few centimeters, and that provides the robustness. Knowing where you are helps then build on top of that the higher levels of behavior that we would expect to see in a self-driving car.

So when I drove around Boston for about a month, I remember one time being surprised, because I drove across the Mass. Ave. bridge, or the Harvard Bridge, from Boston to Cambridge, and to my work at MIT, and they had repaved the road surface. So, what the video that just played showed was, on the left, on a Friday, on the right, on a Tuesday, four days later, and the road surface was completely repaved and the lines were gone.

Now, the Google car doesn't do lane following in the way, say, the Tesla autopilot does, it matches laser scanner data to the road surface appearance map. But it's pretty clear to me that if you suddenly repave the road surface, you need a new map. So how do you maintain the maps to provide the [? very ?] robust localization, the where am I capability? To me, that seems a big question. How do you do that at scale for the whole country or the whole world?

So the question I ask, folks, is, what's the difference between the video on the left and the video on the right? And so here, I have a couple still images taken from roughly the same spot on the Mass. Ave. Bridge. And what you can see is that we have a sunny day on the left, and a rainy day on the right. That's important. And I'll get to adverse weather in a second. But the real critical difference is these lane markings here, they're gone. And so, perhaps a little hard to see in the video, but the road surface was repaved, and we just have these temporary little tag marks to mark the lane locations.

Notice this big truck that drove right across the road, probably a meter away from me, and we're just used to doing that. But the lanes are pretty narrow on the Harvard Bridge, they added some bike lanes about five, six years ago. And how a car would provide the precision localization in such a situation is pretty hard. Now, for vision for the future, I would love to have object based mapping where a robot can think about things like the light posts, and the railings, and the green building at MIT, and use a more semantic understanding of the world, where it can understand the world in terms of objects, rather than just going off the raw geometry and the appearance.

A little bit more on that same example. The next day, I took the T and walked to work. And so I walked across the bridge, and I took pictures, and you can see these temporary lane markings. The point I just want to stress is that current approaches rely on accurate a priori maps of the road surface for precisely estimating the vehicle's position.

Even worse, what if we had snow? So challenge two is adverse weather. And here, this is roughly the same position and those lane markings are under the snow here somewhere. And there's no hope of seeing them with current sensors we have today. And so dealing with adverse weather is a huge challenge. So, with adverse weather rain, snow, fog, they can limit the range of sensors, they might produce sort of ghost readings. The laser scanner-- we did some testing of laser scanners in the rain and it wasn't fun, you know, lots of false readings. It's possible that's surmountable. But if you think about this map issue, the fact that the snow obscures the road surface markings, and you can't see the lanes that are underlying here, that's going to make estimating your location difficult. To me, it could be a really long time before we have a car that's smart enough to sort of follow the car in front and follow just the sort of tracks in the ground, but to do that in such a robust way that it would be safe.

The next challenge I want to talk about is interacting with people. And to me, this is a big challenge. Interacting with people both inside the car, inside other cars, and people outside the car. And so one example shown in this video here is, what happens is-- this is on my commute home at night through a place called Coolidge Corner on Beacon Street in Brookline, Mass. In just about a 30 second video segment, one police officer waves me through a red light, and then a short time later, another police officer stops me at a green light. And if you look at the video and you see the hand gestures, you know, like waving you through and stopping you, how do you write the code for a robot that says, always stop at red lights, unless there's a man on the side of the road who you think is a police officer who waves you through the red light. Or, always go through green lights, unless there's a person on the side of the road who sticks their hand up and it's a police officer and you have to stop and obey him. So that sort of interaction with people, I think, it really exposes some fundamentally deep questions in artificial intelligence that I don't know when we're going to get the answers to them.

Another one of my favorite examples for interacting with people is making a left turn across traffic. So this isn't dash cam data, this is actually a cell phone from the passenger seat recording as I have to make a left turn onto a suburban street where there's no traffic lights. And there's cars coming from the left at a pretty high rate of speed, but there's a lot of traffic, and there's cars backed up looking to the right as far as the eye can see. And so the challenge is, how do you pull out into traffic blocking the high speed cars coming from the left, performing a sort of negotiation with the other drivers where they might take mercy on you and create a gap and let you pull in. And so, you don't see it on the camera, but I gave just a little wave at another driver, and she nodded back to me and created a gap that I could pull in. But, I have this turn on my commute every day, and each day a different surprise happens, you know, it might be a snowplow coming suddenly, and they wouldn't have stopped. Interacting with people in other vehicles and making left turns across traffic are really hard problems for self-driving cars.

And the fourth challenge I call robustness-- robust computer vision. And that's creating systems that achieve a probability of detection that approaches 1, so like 0.999999999, and has a probability of false alarm approaching zero, so 0.00000001.

As an example for this, I'd like to show this picture here. What do you see in this picture? And so, people will say, well, I see the sun.

I see a green light. I see another green light. Now, that would be really hard for a computer vision algorithm to detect the green lights, would be challenging. Now, it's possible that you could use polarization or high dynamic range cameras, there might be ways to attack that. But if you look closely in the picture, and I realize it's nearly impossible to see, it turns out there's a police officer standing right here. You can see his yellow vest, you can see his legs, and you can see the shadows from his legs here. And so that police officer just gave me a little wave, walked across, and started waving pedestrians across. So even though the light's green, I have to know as a human that there's a police officer standing there and he's directing people to walk across my path, even though, for a vision algorithm, it would be tremendously hard to see.

So when we think about the performance of detection algorithms, we typically use precision recall curves, so-called PR curves, or receiver operating characteristic curves, known as ROC curves. It's important to remember that no sensor is perfect. And so if we think about this probability of detection and probability of false alarm, that's a very important concept. And these concepts go back to sort of World War II, or earlier, looking at the performance of radar systems and sonar systems.

And what I want to do next is just say a little bit about this notion of how you assess a sensor's performance. And typically with a sensor or detection algorithm, you have a threshold. And part of the art of engineering one of these systems is choosing the right threshold values for such a system.

So now what I want to do is talk through two different examples for an automotive situation. So first, I want to talk about how you might design an active safety system. And I want to credit Chris Armstrong for Google from this analysis, this way of thinking. What Chris did, which is really kind of cool, is, you want to convey to the layperson what some of the issues are. And so, he gave a talk in Pittsburgh earlier this year where, instead of talking about probability detection, he said vehicle avoids accidents. So suppose it was pedestrian detection, people walking around looking at their cell phones, they walk in the road when they shouldn't, maybe there's a person in front of them is walking across a "don't cross" and they're just following the people around them and they step into the road. And so, what you'd like is very high performance in detecting, say, a person stepping out into the road. But the false alarm axis, Chris rewrote it as a vehicle stops unnecessarily. A false alarm would be if a paper bag blew in front of the car and it stopped as if it was a person.

So imagine if you're driving down the road, you're drinking your coffee, maybe 35 miles an hour, and suddenly-- [WHIRR]. The car stops for a paper bag. And you're like, what's

wrong with my car, that's not good. And so, if you're designing an active safety system-- so for example, Diamler Mercedes have given public research talks where they talk about their goal as to achieve sort of 80% detection-- let's just pretend 50. So let's say-- we'll say 0.6. Let's just say 0.6 detection and with a very low false alarm rates, so say it's like 0.1. OK? So an active safety system might have a curve like this where you have to choose the thresholds so that-- you don't want too many false alarms. You don't want your car stopping unnecessarily when you didn't expect it. But, at least with current technology, to get that level of performance, that means getting a detection level that's pretty low. And say if it was 0.6, that would be like three out of five.

So imagine you had an active safety system that three out of five times, if a person jumped into the road, it would stop the car, slam on the brakes, prevent an accident. That would be great actually, in some measures, because it's just meant to augment the human driver. And what Diamler would say is, well, for the other situations, you know, it's really the fault of the human driver for not paying closer attention. However, for an autonomous car, this might not be good enough. And an autonomous car might need something that goes like this, because you may need to choose a point that's really, really high on the detection axis, because there's no person to jump in and prevent that. And so that might mean a higher level of false alarms, the car being sort of timid and not going sometimes when a person would say, just go. And so, you have to choose this trade off between how brave you are versus how cautious you are. And that really maps right back to the basic physics of how the algorithms work, how you do detection and noise or classification with very challenging and noisy images.

So my view is to achieve fully autonomous cars. It's actually a long term research issue to get to this point, to get to the place where you get a very high probability of detection with minimal false alarms. And it might take us a long time still to get there.

Robots and Autonomous Vehicles (John Leonard): Video 7

I want to review the NHTSA levels of autonomy that are defined by the government talk about different types of self-driving vehicles. And so level 0, no automation, is where the drivers in complete control all the time. Level 1 is where there's automation of one or more specific functions. This might be, say, cruise control.

Level 2 is a combined function automation. This might be adaptive cruise control or say there's a radar looking out the front of the car and it controls the braking in traffic as well as a traditional cruise control system. So in summary, with level 2 the driver has to pay attention all the time. They are responsible for the safety of the vehicle.

Things get more interesting with level 3 and level 4. In level 3, the car does most of the driving, but the human has to be available to take control once in a while if the car needs help.

Ideally, this would be with sufficient notice to give the person warning and say, in 30 seconds I'm going to need you to take control of the vehicle. But it's possible that that control might be very rapid. It might say, suddenly, I'm confused. I'm looking into the sun. Take over now.

And so that level 3 partial automation can be contrasted with level 4 full automation. And level 4, the human would give sort of a high-level objective, take me home, but then they could go to sleep, they could read a book, they could work on their laptop. In level 4, the car has 100% of the responsibility for the safety of the vehicle.

As we go forward, the difference between level 3 and level 4 is critical. Level 3, it's pretty much here today. With level 3 you need the human to pay attention to take the hand-off when it's necessary.

The problem with level 3 is that humans are really bad at monitoring autonomous systems. Suppose that the car does a good job 99 out of 100 times. And imagine if on your daily commute the car just it has it, it is doing a great job day after day after day.

Well, humans will let their guard down and they'll assume that the car does better than it really can. And then on that 1 out of 100 time when the car needs you to take over you might not be paying enough attention to be ready for the hand-off. So partial automation brings in this hand-off problem which is a very difficult issue in human-automation interaction.

Level 4, however, there is no human to take over, so we need to achieve a much higher level of robustness in the perception algorithms. And so it's not good enough just to sort of help say 60% of the time. For that pedestrian we need to get it 99.999999% of the time. And so the challenge for level 4 is robustness of the perception system and the overall integration of the system.

So in summary, we have this big fork in the road. Is it level three, partial automation? Is it level 4, full automation? And to me these are the big questions going forward. So for level 3, partial autonomy, can humans be trusted to take control when necessary?

For level 4, can we achieve sufficient robustness that the detection probabilities will approach unity with very low false alarms. Because there is no human to jump in and save the robot, it has to do it all itself.

Robots and Autonomous Vehicles (John Leonard): Video 8

So in summary, it's an extremely exciting time to be working in mobile sensing and SLAM. Location information is so vital for a wide range of industries. Engineers that can build real-time 3D perception and navigation systems and combine them with motion planning systems are in really high demand for industries such as virtual reality, mobile devices, self-driving vehicles, and drones.

And additionally, the connection to the Internet of Things offers a much broader set of opportunities spanning many industries. Knowing what is where in the world, knowing where you are is just a vitally important technology.

My vision is that these technologies for robot sensing can be applied to a much broader set of challenges in industry. Anywhere that location information-- knowing what and who is where in the world as a function of time-- to me, that has to have economic value. And so I think that if we can bring this notion of location and sensing into the physical world, it's going to have a broad impact.

I do think there are some great research challenges still to pursue. And I'll go back to one of my earlier slides, why is SLAM difficult? I talked about these different axes of representation, inference, and systems and autonomy. In each of these areas we have exciting things to do.

In representation, I think we really need to represent the world in terms of objects. We need to take advances, say, in deep learning approaches to object recognition, and use deep learning as a sensor where we can infer what is where in the world and give labels to things.

I want to use semantic representations to try to get to a higher level understanding of things and try to reason, ultimately, about behaviors. And I think we also need to pull in text, the notion of capturing all the text in the world, and coupling that to these dense 3-D representations. That will enable more intelligent interactions. That will enable home health care robots, medical robots. There are a range of applications where rich dense models of the world can be very helpful.

All of these techniques for representing the world feed back to new challenges in inference. And particularly, I think learning is vital, not just traditional machine learning where you train a classifier for a particular data set. But I want to see learning on steroids, lifelong learning where you can really think about the limit, as time goes to infinity, how does a system get better and better and learn more and more about the world?

And ultimately this entails connecting to the cloud. When one robot learns a Coke can, every robot should know what a Coke can is. This notion of sharing information, things getting logged to the cloud. I have this notion of a robot that operates autonomously each day, capturing new experiences. And then at night when it goes back and connects to charge its batteries, there's a sort of dreaming that happens overnight, of trying to make

sense of all the data of that day and connect it to the data previously acquired by itself and other robots to try to build ever richer and deeper understandings of the world.

So combining representation and learning and inference, ultimately we need to create a robust systems that are autonomous. And this is how we can create systems that we can truly trust to operate with people that will be safe, that will be efficient, that will learn from their mistakes and get better over time, and be able to use real-time closed-loop feedback control system to actually control motion through the world.

So in summary, I want to say it's a really exciting time to be working in robot sensing. And I want to thank you for your attention and wish you luck in taking these technologies to create an impact in the world.

Week - 4 Module Four: Applications

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 1

Hi. I'm Joe Paradiso so from the MIT Media Lab, and I'm going to talk to you about how the Internet of Things is going to touch people. Once we have sensing rolling out across the whole planet, my research group is very, very interested in how people connect to this electronic nervous system in different ways. I'm going to be giving you a lot of examples from my group's research throughout the talk, and if you want any more information, a lot of it is posted off our group's website at the bottom of the slide. I'm going to take you guys on a tour through many different topics looking primarily at different applications for Internet of Things that we've been exploring.

First, I'll give you a little bit of background on the field in general, what we call ubiquitous computing, which, in my opinion and the opinion of many of us, is really the parent field that Internet of Things came from. So I'll give a little bit of history and projection there. I'm going to also talk about emerging data standards that are going to really enable the Internet of Things. I'm going to talk about a little bit of work going on industry-wide, as well as highlight some work in my group, looking at standards for generic sensor description.

I'm going to talk about also something that's very, very close to my own interest and really the research interests of my group. It's plugging in viscerally to this network of sensors that are everywhere with immersive visualizations, really treating the universe of sensors as a nervous system that we can plug into and extend human perception through. And there are a lot of applications that come off of that.

I'll talk also a bit about the work we're doing in Internet of Things in the Built Environment, systems that leverage Internet of Things sensors to lower energy costs, handle lighting, so on and so forth-- a huge application for this field. And the interface of the future will be, in many ways, on-body, both on-body and infrastructure. So I'll talk a bit about that as well. We've done a lot of work over many, many years-- over decades-- at the Media Lab on wearables. And I'll discuss some of the things we're doing now.

It's interesting if you think about the evolution of tools under Internet of Things. We always use tools to do tasks. They become an extension of our body in many ways if we're practiced using a tool. What does Internet of Things do to that when the tool becomes smart? And I'll give you some examples there that we're looking at. And finally, I'll talk a bit about sensate materials. If you think about Internet of Things on surfaces on materials instead of just in rooms, buildings, and environments, how does that really change the material? What kind of material can you make? I'll talk a bit about that as well.

And finally, I'll give you pointers to resources. Throughout this lecture, I cite papers that my group has published. I recommend, if you're interested in any of these things, take a

look at those papers. Read about the prior work, read about our stuff, and you'll learn a lot.

This is my team of students at the Media Lab, the Responsive Environments Group as of last spring. These are the people that did the hard work, for the most part, that you're going to be seeing in the talk. And it's a wonderful group of people. Great computer scientists, great engineers. And, at the Media Lab, a lot of our students are also artists in some ways. In my group, half of them tend to be musicians, so it always makes for a lively get-together.

So let me start with the enabling principle, in many ways, that allows us all to be here and talk about such wonderful technical revolutions. It's Moore's law. It's the principle that drives, essentially, the advancements in electronics at its heart. And it's a simple scaling law. Gordon Moore at Intel came up with this during the '60s, just looking at empirical data from his industry, coming from Intel. And it's saying, essentially, that the density of transistors doubles every two years on an integrated circuit.

And this has held remarkably steady for decades. We've been able to rely on it to know that technology is going to be getting better, that your devices you have now will be obsolete in a few years. You'll need new computers, new phones. There's going to be a revolution in capability. Of course, Moore's law is ending by the end of the decade. We don't know how we're going to scale beyond the quantum limits we're hitting now. And this appears to be a very serious roadblock. And we have to be very creative to get beyond that. But we're already in Internet of Things.

Moore's law goes through epochs. And, if you take a look at, for instance, Gordon Bell's waves of computing-- Gordon Bell is a famous computer scientist who came up with many great observations and lots of great advancements. But he made this observation that Moore's law, as it increases capability over the years, passes certain thresholds where certain applications are possible. And, looking at computing, you have what he calls different waves of computing that get enabled, essentially every decade.

So, starting back in the '60s, we all had mainframe computers. And then we evolved to minicomputers, when I was a kid in the '70s. Then we went to desktops-- PCs. We all had those. Then we moved to laptops. And now we're really in the age of handhelds-- phones and tablets. We're really midway through that era.

So, the question is, what comes after? And that's really Internet of Things-- what I call an everywhere computing. I'll soon give many names for it. Essentially, it's ubiquitous sensor data streaming into the cloud. It's going to be everywhere. And then context will project down into the user. This will be the user interface of the future. As we wander around, information about us will project up-- from everything, not just from your phone, which is what it does now-- and context will project down to cause things to happen.

The interfaces for this as well are going to move off of the platforms that we have now-- the tablets, the phones-- and really onto the body as wearables-- many of us firmly

believe this-- and into the infrastructure. So, it'll be sensors distributed everywhere, including sensors that you're wearing and displays and actuators that you wear. This is what is going to define how Internet of Things touches people, through this kind of an interface.

Internet of Things is a vision with many flavors and aliases. And, over the course of my career that I've been involved with this, I've gone through many, many different paradigms and names. But they're all essentially looking at the same kind of a thing-- this revolution of technology moving into everything around us. And really, the beginning of it was Xerox PARC, the work of Mark Weiser in 1990, ubiquitous computing. That started, for many of us, the real flavor and the fundamental themes that are involved with Internet of Things.

And then, of course, everybody jumps on the bandwagon, gives it a different name, a slightly different twist. IBM, in the European Union, coined "pervasive computing," which kind of stuck for a while. It's an alternative name. Microsoft, in the European Union, had the "disappearing computer." Same idea. Computer disappears into everything. "Invisible computing" was a University of Washington and Microsoft term coming from the early 2000s, more or less. "Ambient intelligence" as well, around the same time. Same kind of idea. Intelligence is all around us. Projects down.

At MIT, we've had a bunch of projects and programs that we've unrolled over the decades that have addressed this kind of a vision. At the Media Lab, the Things That Think program is a program I've been very involved in since it started in 1995, really looking at this whole rollout of ubiquitous computing. And, of course, the Lab for Computer Science "Project Oxygen" and the AI Lab "amorphous computing"-- lots of wonderful work were done under those themes. And, finally, in collaboration with the Procter & Gamble and the MIT Auto-ID Center, the term "Internet of Things" was coined. And that's, of course, the reason why we're all here talking to you today.

But let me go back to the beginning and give you a little bit more about what happened at Xerox PARC around 1990. And I give lots of talks these days, industrial and academic talks, about Internet of Things. And I'm amazed at how few people remember Mark. And it's so critical that he's called out, because he's really the pioneer for this whole vision in many ways. Xerox PARC invented many things. They invented, of course, the personal computer. They invented the ethernet. We can go on and on. But they also invented Internet of Things in many ways.

Mark Weiser started a program when he was there, eventually as Director of Computer Science, called ubiquitous computing, where they looked at this vision. They elucidated it. And they actually built pieces of it out with technology of the time, with electronics diffusing into everything, computation diffusing into everything. Look into user interface coming off of desktops that were ubiquitous then and laptops into different kinds of objects, from tablets to things on the doors in the offices, badges you wore.

It was a very rich environment, both intellectually and in terms of the technology they produced. And Mark, who left us all too soon, was the guy who started a lot of this. He wrote a seminal article I recommend to everybody called "The Computer for the 21st Century" in Scientific American in 1991 that really set the stage for a lot of research that happened after in this area. It really captures a lot of these ideas, plus, important to many of us, the work in HCI talks about how this technology touches people. Weiser was all about that during his career.

So, I'm going to start getting a little bit technical and talking about some of the emerging data standards for Internet of Things and for sensors.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 2

Now I'm going to talk a little bit about emerging data standards for the Internet of Things, and for sensors. There's a tremendous amount of work certainly happening here, because it's so critical, but I'll touch on a few things. It's very much something we need. This is a slide I've had in my deck in different forms for the last 10 or 15 years, maybe.

And for us working in ubiquitous computing, it was obvious this was going to happen. Sensors follow corollaries of Moore's law, they get smaller, they get cheaper, it's easy to embed them into different kinds of devices. Even if you don't know what the sensor is going to be good for now, you can put in the device and then develop around it afterward. The phones pioneered a lot of that, but everything now was following in those footsteps.

So sensors are getting embedded into objects all around us now, already. These objects, more often than not, want to be able to phone up to the net, to talk to the net, to connect to the network for their own benefit. Just to cache information, pull information down, store things, connect to the cloud, and its Internet of Things in many ways. But right now, we're living in a world where it's all pretty much balkanized. All the devices around us will talk to a network if they're connected, and more often they're connected, they have sensors.

But they only talk for their own purposes. They don't really talk to each other so much. They just talk to the network, and they perform the function better. They talk to their own apps. This is all going to change immensely in the next few years, once these protocols are in place, to be able to share data across families of devices.

Then we're going to be living a revolution. It's going to be even more powerful than when the web hit PCs. Because when we had PCs around, we could send email, we could have TP files, and some of us did a lot of that. But once you had

Once you had HTML and sort of developed this wonderful open standard in the web, suddenly we're living in a different world. These computers could seamlessly exchange information and data in a very natural way. Browsers were developed. And suddenly within a few years, the world changed. I don't look any more at this machine, this computer, in the same way as I did before.

So we're going to see the same thing happen with everything. And because it's such a bigger scale, it's going to be just as profound, if not more profound. In my opinion, it becomes a phase change. Because we already have a saturated world. We've got sensors. We've got connections.

More and more connectivity. That will grow, the sensors are there. And once we have the protocols to exchange the information, boom, suddenly all these applications will be enabled coming out of nowhere. We'll change our lives very much.

But of course, we need these standards. There are a few coming from industry. These are two of the main ones that I've been interacting with. AllJoyn is a protocol coming out of Qualcomm. And essentially, it's to enable devices that are around to all talk together. Another one is from Intel, the Open Interconnect Consortium at Intel. That's called IoTivity.

And these standards propagate all the way to the device. So they're not just standards that live on the web. They really live on the devices themselves, and through all the layers of connection. And they tend to assume-- for the most part, this is changing-- but they tend to assume a somewhat heavy device, where you have an arm class processor, a more power-hungry processor, than you would have in a little sensor node buried somewhere that runs on the battery, which is the kind of thing we do a lot.

So for our work, this is all wonderful, and this is going to certainly have impact. And there are other standards as well. These two are the main ones.

But in our case, we needed a standard we could use right away that we could to describe sensors in very agile framework in a very agile way, that we could roll out now, and that would scale to low power sensors. So what we did is developed this protocol we called CHAIN API.

And it's a very lightweight, adaptive way to describe sensors-- real sensors and virtual sensors. We can actually have aggregates of sensors posting CHAIN as another sensor. And do it as a RESTful standard. It's written in JSON, a natural web language. And it allows sensors to specify the characteristics, their data. And in addition, links between sensors coming to other sensors through this.

And it's by nature entirely decentralized. So the sensors don't have to live in anybody's domain, in anybody's cloud, in anybody's server. The sensors can be distributed all over, just like the web naturally is. And again, they can link to one another. You basically aggregate and pull the sensor data really by crawling this web of sensors like you crawl the real way.

And that's how we see the future of all this becoming. CHAIN actually deals with lightweight sensors by basically ignoring them. Because I believe in the future, sensors are going to be sending data to a variety of different protocols, because they're going to be having different resource constraints.

So a battery-powered sensor that has to last for 20 years has a very limited energy store. It's not been able to be verbose and speak web literate languages so easily. So to speak their own really tight protocol. But once they hit the internet and the web, then they can be interpreted and posted in something of a verbose-like CHAIN.

So we've foreseen this. We leave this off. We let our own lightweight protocols work, and there are many of them that refine. We develop some. Other people develop some.

But once it hits the web, that's where you want it all come together. So then we translate into CHAIN, and we can speak it, and crawl it, and work very naturally with it.

As well, we don't need to deal with databases in any particular fashion. We use databases. We don't demand a certain database format. You can pretty much have any kind of database you want that works. It just has to post the data and respond to CHAIN queries. So we just basically have a front end to the databases and to the sensors. CHAIN lives in between.

This is an example of sensor data in CHAIN. If anybody wants to know more about it, you can go to the GitHub site. It's public domain. And you guys could play with it, use it all you want. Again, it's JSON-based. And the sensors basically declare something about themselves. They link to sensors that are nearby.

They link to parameters that define what these sensors do. Limits, so on and so forth. And you can even link to a different kind of data, like a stream. You can subscribe to a sensor and CHAIN, and have that automatically push data when it updates at the rate we desire. So this is all capable in the CHAIN protocol. We also have a paper, and a MobiQuitous that describes it.

So now I'm going to move on to applications. And my group, most of our data is posted in CHAIN and so we built lots of applications that exploit CHAIN to visualize and process data in different ways. I'm going to give you some of those examples now.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 3

Visualization of sensor data, especially immersive visualization, is a very interesting topic to us. How do people connect to this world of sensors? How does it almost extend human presence, and really play with the feeling of what is here and what is now? It really fascinates us in the Media Lab.

My student Gershon and I wrote the cover article, actually, of Scientific American last July, a year ago last July, really talking about what this is like and illustrating a few of the projects in our group that I'll go into a bit more detail about here. This is the Media Lab, beautiful new building, all made of glass. You can look through and see everything that's going on. Lots of light, it's wonderful.

But let's say you could look through with eyes that could detect sensors. Maybe this is something like what you'd see, because buildings have their own autonomic nervous system to control the air conditioning, the heating, and the lighting. What if you could see all the sensor data coming from that and bring it together and visualize it?

And I think this is going to be very much a part of the Control Panel for internet of things in the future. Because one of the fundamental things about it, is that sensors from different devices will control different applications. A sensor for the HVAC is not just going to be used for the HVAC. It could be used for something else. The sensors for the lighting could be used for something else, for security. They're all going to be mixed in many different ways.

And I think these device agnostic, or sensor agnostic browsers, are going to be really key to understanding a lot of what's going on to visualizing it, in addition to the way it pokes it at the sense of presence. So we've done that with the Media Lab. Some of it can be seen if you go to the website doppelab.media.mit.edu. You can run a version of it there. But we'll try to run it now in real time.

This is all written in Unity 3D. It's a very common game engine. People have written great games in Unity. Here, we're using the game engine to interact with and explore real time data coming from sensors. So now we're looking at the Media Lab building. We've imported a CAD model into Unity.

I could turn the walls on and off. That's with the walls on. Here they're translucent. And we can see a lot of things going on in there.

The little flames that you see all over the place, the kind of orange-ish or white-ish flames, represent the temperature in every office. We use the flame as a metaphor for temperature. If it's reddish, it's hotter. If it's bluish, it's colder. So you're seeing how the whole building is being heated right now.

You see these big spheres around some of the flames. That's a flag representing the fact that the thermostat in the office, the temps in the office is not meeting the set point that's

defined by the thermostat. So if it's a red flashing sphere, it's too hot. If it's a blue flashing sphere, it's too cold. So those offices aren't being regulated properly.

The red ones, for the most part, are server rooms. Though that changes, some offices just get hot. The blue ones, I don't know. They're at the beginning of the HVAC duct or something. They're just always too cold.

And then if you look over here in the center, you see a bunch of sensors that we put in the main atrium. And they're measuring the temperature humidity with dense spacing every meter or two, in the large open atrium that we have at the Media Lab.

We did this as a project with Schneider Electric. They were sponsors of our group. And we designed these. We put them in with them. And we did studies of air flow in large atria with MIT Architecture in Building Technologies.

But now that we have the sensors there, we just have it coming into DoppelLab, because DoppelLab is a universal way to show data from these sensors. So you can see today, it's raining. It's humid. And the HVAC isn't really doing a great job of sucking the humidity out of the atrium. You can see the red fog here. That basically means that it's quite humid in the atrium. It's a little bluer up here in the corner. And that means the humidity is lower. So the HVAC vent up here is actually starting to suck the air out here and dry it out.

The temperature is the color of the different objects here. If it's reddish, it's warmer. If it's bluer, it's colder. And some of them turn into arrows. Here the temperature has been going down. Here the temperature's been going up. So we can see the behavior of temperature and air in this large space in real time through DoppelLab.

So you can see these green streamers. Here's one here that's moving up and down. That means people are moving. They're motion sensors that we have put ourselves in the common areas of the lab. And when this moves up and down, people are walking by. And when it gets longer, it's louder-- shorter, it's quieter. So now it got a little loud when someone walked by.

Of course, we have microphones, and we can hear these sensors if we want so. Now as they get closer, I'm hearing audio from the microphone itself. It is at this location. It's all spatialized. This is real time audio with a seven second delay. So we can hear activities like the elevator going off, people moving around. You can even hear people talking, if there are people there at the time.

But we don't snoop on anybody. This is paramount. We don't want to even go close to that. So right at the center node itself, we distort the audio. We reverse grains of audio. We divide the audio first into grains. We reverse some of the grains. We mix grains. We drop grains. And we smooth it back out so it sounds kind of natural.

It sounds to me like people are speaking some very strange language. But clearly you can hear them laugh, you can hear if there's a talk going on, if there's a group of people. You get a weird feeling of presence once you have the audio here.

We can also go back in time. So I can go back to last night, if I want. So this is 12:00 AM. It was much drier in the atrium, as you can see. So it's blue. It hadn't started to rain so much yet.

And the building's in set back. It was colder out. It's approaching the Boston winter now. And these offices are cold because the building isn't set back. So you can see it's a totally different operation.

And if we want, we can go back further in time. Let's go back to when we had one of our large sponsor meetings. So I can speed time up. And we used to have the audio stored. We don't really have the audio stored now. But we stored all the other data.

And this is the time we had a lot of our industrial partners in the lab for our semiannual meeting. And they wear RFID badges when they come to visit us. So as they start to come in-- you can see how fast the clock is going here-- you can start seeing them all piling up, coming into and going out of the sponsor meetings. And this is using standard RFID distributed throughout the Media Lab.

There's somebody. And there are more people. We actually catch tweets at that time, too. So of people tweeted anywhere in the world, tweeting as a virtual sensor. So there are people coming in. Again, if we have their photo, if they publish their photo with us, you can see it. Otherwise, you see an icon on the tube that corresponds to their location.

So you can see how the building is being used, as well as how it's reacting. So with a tool like DoppelLab, it's already useful for people who maintain large facilities. Because if this is a large factory, if it's an oil refinery, if it's an oil drilling site, anything of this sort, you've got sensors all over the place already. You have lots of assets. It's all geo-located. Everything has a location tag with it.

So you want to be able to see in real time, how are things working? And again, even a building, buildings are complex systems. This is already useful to see how the building is reacting to things that are happening.

So the sensor agnostic browsers like this already have a use. And going forward, under the internet of things, when the number of sensors and the amount of data explodes, this will be very much a part of how we interact with these systems.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 4 Part 1

So, this is a video of DoppelLab showing it in operation, showing a few of the features. This is, again, the atrium. This was in the atrium when you're flying through. And it was a dry day there.

Now you're going up the atrium, flying through. There's no laws of physics, you can just fly around. There is one of my students walking by that RFID reader.

These are there people walking by, you can see motion in the sensor. You can start to hear them in the audio. You cannot understand a word they say. And, of course, audio from anything is fair game for this kind of a framework.

And during that time I had my synthesizer-- I'm a synthesizer builder. I built one of the world's largest modular synthesizers over the years. I had that at MIT and it was streaming live audio. So now we are hearing what was then live audio coming from my device.

Once you have all of this data flowing into these animations in real time, and including the audio, you have this spooky feeling, in a way, of being in a weird dream where you're kind of somewhere, but you're kind of not. This will get more vivid as time goes on, but already hints at redefining presence, something that really fascinates us.

So for our next project we went, in a way, further from application in some ways, and closer to application in other ways. But really motivated by this whole idea of using lots of sensors to play with presence.

We went to a cranberry bog in Southern Massachusetts. This is in Plymouth, it's about an hour's drive south of MIT close to the coast. One of our retired faculty, Gloriana Davenport, and her husband co-own a 600 acre cranberry farm that at one point made up to 1% of Ocean Spray's cranberries, the major manufacturer in the US. It's no longer economical to grow cranberries here, for many reasons, so all of the old cranberry farms, for the most part, are getting retired.

So in this case, Gloriana and Evan wanted to restore the whole property back to nature. And as Gloriana made her career here as faculty of the Media Lab years ago all about the future of documentary, she became fascinated with the idea of having the sensors themselves produce a documentary of this large site going back to nature.

So we're collaborating with people in environmental sciences at various universities around Boston to put sensors in there. We're designing them ourselves. This is the restoration plan. It's now Massachusetts' biggest restoration site. They're completely changing the way water is flowing and the way things will be collecting and growing in the space.

And we have built, as I mentioned, sensor nodes to measure some of these parameters. This is the basic workhorse node that we have made.

And I want to stress here that micro-power sensor nodes, somewhat like this, are going to form the leaves of the trees of IoT. IoT is very much about networks, very much about data, but the data is going to come very much from these small battery powered sensors that are going to be everywhere. Low power sensors embedded in environments that make measurements and post when relevant. This is going to be an important part of IoT. And we've developed a sensor node to address these concerns, address these opportunities, in our Tidmarsh project.

This is a node my student Brian Maiden has made that measures a whole bunch of basic environmental parameters. So it's temperature, humidity, light levels, air pressure, motion, and audio features as well, we have microphones we can put on, we can process the audio.

And it's easily expandable. So, this node is built to be expandable, built for sensors to plug-in. It measures soil hydration, it measures wind, it measures air quality, things like that.

And, updating about two or three times a minute, these nodes will last about two years on a set of AA batteries. If you put a solar cell on it, lasts essentially forever.

These are a bunch of nodes waiting to be deployed. We've made about 100 so far. We're talking about going up to over 1,000 this year. We have industrial partners that are going to be coming into this project.

And we've made our own network for this. It's essentially a 802.15.4 standard. It's one of the very common wireless low power standards. There are lots of things that live on top of that. ZigBee is one of the most famous sensory networking standards that lives on top of 15.4. We've modified ZigBee a bit to make our own standard that fits better with our application. But essentially that's where it came from.

There is more information about the sensor node at the URL below if you're interested.

The sensors all connected via 802.15.4 to a base station. This is the base station here in one of our insulation areas, we have several of them now. This base station will pull in the data from the sensors and then send it through directional WiFi to a barn about a kilometer away where it hits the network itself. So the data is collected here.

Again, there is no power on site. Everything has to come either from embedded batteries, or from solar cells, or energy harvesting sources in other ways.

This is my student Brian Maiden. He's the main architect of the sensors, pointing to one of them that's deployed there. The rest of the team and some of our collaborators there visiting Tidmarsh in the early summer, late spring.

This is students visiting in the fall, here's a node that has soil moisture probes. And this is just another node by our marsh. Again, we have them all over.

Here's a node with a solar cell that's a router. So it can route packets, stays alive. This node has a weather station. So, again, we can configure the sensors as desired for different nodes.

This, of course, is what happens if you aren't careful enough with nature. We left a tiny little seal off of one of our sensors, only a seal about maybe 3 millimeters in diameter, but the ants found their way in. So we're measuring an ant colony instead of the other aspects of the environment.

If we go to the website, tidmarsh.media.mit.edu, you can look at the data in different ways, read about the project, see all the different sub projects we've done within the Tidmarsh facility, as well as download an app to browser the data yourselves.

I'll show you very quickly here, if you look at live data, we can see data live from one of the sites. It's raining now here in Boston. In Plymouth I believe it's still raining. This is a live image feed updated every so many seconds from Tidmarsh.

You can hear sounds coming from the microphones we have embedded in that site. And, of course, we can look at the sensor data as basic plots that we're collecting here. This is all wireless solar cells. There's no power.

This is another area here. You can see there's water on the lens, but that's an image, a real image. And, of course, we have sensor data that's been collected over the days. See the diurnal cycle.

And then, get another site. That's a greenhouse and sensor data. You see where the sensor is located at the bottom in all of these plots.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 4 Part 2

And of course, we can also take all this data, import it into an immersive game engine visualization, just as we did with DoppelLab. And we've done that with tidmarsh. We've had the topography laser scanned and we've reconstructed it. Again, we don't yet have the rain clouds, we're working on that.

Also the rendering that you see here is kind of a late-- early fall rendering. We're going to have the rendering update automatically with cameras on the side. But you start to see the sensors now appearing in the game. This live posting, maybe seven seconds or so delay, of actual sensor nodes on the tidmarsh site.

At this point they're displaying the temperature. And you know, we can move, it's a game. We can even jump if you want.

So we're seeing the sensors update in real time. We're seeing temperature, we can look at the humidity, we can look at light levels, we can look at all the data together. That's a barometric pressure. This is all the data together here. And you're hearing also the sensors make music.

So you're hearing audio. The audio right now actually is cached audio recorded in the spring, that's why you hear the birds. But I can hit a key now and you're hearing live audio from the site in the background.

But the music is probably dominating this now. And the music is coming right from the sensors. And I see these environments, these rich data environments, that are coming from Internet of Things very, very quickly, is it not only source of new applications and new opportunities for businesses and all of us, but also as a campus for future artists. Because this is a very rich source of information that reflects a real environment. And a creative mapping of that kind of information on artistic content, I think is really profound.

So in this case, we have taken data from these sensors we've mapped it into music. The sensor just updated. And-- the pluck when the sensor updates is coming from the humidity, the tambour comes from the humidity. The pitch of the drone as I move around, is coming from the temperature.

So just from the sound that you're hearing, I'm experiencing the conditions in this cranberry bog, this faraway place. And in some sense, you know, I mean those of us who play with virtual augmented reality love the phrase better than being there. And you know, it's going to be a long haul before we get even close to that.

But you know, if you go to the real tidmarsh now you're going to get soaked. If you go during the summer, it gets hot and sticky and it's infested with ticks, we're all going to get Lyme disease. This is another way to experience it.

So what we're doing as well, is we're opening this framework up to other composers. So we're opening up tidmarsh and some of our other frameworks of large data stores, real time data, to composers building a bunch of tools, so they can easily author music and put it on top of this data. So you can experience this cranberry bog in the mode of name your favorite composer.

And it will go on forever. Every time you visit it, it will be different. And it will change with the environment, as you move around, could be a city, doesn't have to be a cranberry bog.

So this is a video that shows the tidmarsh project, the virtual reality version of it. So sensor data was pouring in live, you're seeing it update. The position of the sun changes also according to the time of day.

And my students made this piece of music. I actually played for them some of my favorite ambient pieces, music for airports by Eno and Wendy Carlos's Sonic Seasonings and they kind of got in the mold. And they nailed it beautifully here, so it's kind of comfortable. It's a little bit aesthetic.

And the background is the real audio that was happening at the time. Of course at night, everything can change completely, including the music, right? Because evening is a different kind of an ambiance.

So when it gets dark, it's a little bit creepier. And you hear the frogs. Depending on the season, you hear lots of different wildlife there.

[FROGS CROAKING]

So this is a different aesthetic experience, but more suitable to a night out on the bog. We have many other projects running under our tidmarsh infrastructure. In many ways it's a large test bed for us to explore different internet of things enabled opportunities. And one of which is thinking about a mobile agent.

So we have fixed sensors, you saw that. But what if you have a mobile sensor in that environment that you can move around? What if you have something like a drone and you could fly the drone around from within the game?

Just as you move your first person perspective in the virtual world, you could move a real agent in the physical world. I'll do it from within the game engine. It really nicely unifies control and sensor data and interaction.

So we built a framework where we can fly a drone from within the Unity game engine, within the tidmarsh app. We can be in the game looking up and see the drone and all the data. We could be in the drone with real video, looking out and seeing from the drone's eye view. We can be half and half and see the data coming from the game's visualizations on top of the video coming from the drone.

So my student Vasant has built this framework called Quadrasense to actually do that as part of his thesis last term. So this is a video describing Vasant's project. So he has an omnidirectional imager, it's a fisheye camera that's real time rectified, streaming right off the drone. Video quality is actually better than what you see here.

He controls, as I mentioned, the drone completely from within the Unity game engine, just as if you control your first person perspective. So there's the drone over the real tidmarsh, that's the virtual drone there. And here he can look out from the drone's perspective. And he's looking at the game views. You're seeing the data, you're seeing the animated landscape.

Now you're seeing the real landscape with the animated data. He's got some lag issues he's fixing now, but it's starting to work. So we think also lot about being on such a site with what we call wearable sensory prosthetic, where you could, for instance look across the cranberry bog and hear through your wearable sounds from across-- the microphones across the water. And then look down under the water and maybe hear the sounds underneath. If you have hydrophones or microphones nearby, you can hear under the water, let's say, right?

We're planning to do things like that at tidmarsh. We already have plenty of sensors there. So we're building such a device.

This is current research in my group. We have built a immersive spatialized audio interface, where through bone conduction essentially, it gives you a second set of ears. It doesn't cover your ears, but you can hear clearly through the bone conduction.

We tracked the head completely, so we have a full head tracker on it with inertia measurement unit and a compass. And we also locate the headset. We use GPS to locate it coarsely, and if you put in a more precise location system, like ultra wide band, which we have enabled on this, we can locate it to within some centimeters, depending on resolution. So you know fairly accurately where you are, you know where your head is pointing, so we can start you know, spatializing sounds that come from the sensors when you're there.

The sensors themselves can make the music, make sound, microphones that are embedded in the environment can produce sound and you hear them from where they should be in the audio field. You move your head around, they're going to be fixed in position. So, we've built this, we've tested it indoors and at MIT. We're going to be testing it at tidmarsh shortly.

Going further, we want to begin to vector off attention. What am I paying attention to? That's going to be key to the user interfaces in the future. They're all going to respond, not to clicks on the screen so much, but more to what you're paying attention to and giving you the right information at the right time. So, we're starting to explore that very hard problem.

We can start to make some progress. With a bunch of sensors we're putting on there for that, certainly EEG, very core sensor, but it helps. Pupil diameter, eye direction, gaze direction, can help. EMGs as well can see how you're kind of moving your muscles, where you're focusing.

So we want to make this like a natural, again, sensory prosthetic. You're paying attention to something, you start hearing the sounds that are coming from that, seeing information if there's a visual overlay from sensors that are embedded there. Make it natural. Keep your hands off a screen, off a device because that's going to be a last resort in this world we're moving toward.

You know, my students at the Media Lab didn't want to wait for everything to be ready at tidmarsh to experience this idea of a sensory prosthetic. And in particular my student, Gershon, who's an artist as well as being a great technologist, wanted to really experience the whole idea of hearing remotely just in his day to day environment. So he, on his own, built what he called a sensory prosthetic that were a set of parabolic microphones that he mounted on his head. So, this is a wearable.

He's got two parabolic microphones. He controls them via joysticks. And he can point his hearing, essentially wherever he wants. And it certainly does extend your hearing by quite a bit. These are bona-fide parabolic microphones like people use to hear remotely.

So even at the Media Lab, it's catching some notice. And the snow of last winter.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 5

So now I'm going to switch to some work that we've been doing in internet of things and the built environment. And this, I think, is one of the largest recognized applications from internet of things, is regulating energy use, lighting, so on and so forth, inside buildings. Already, businesses are rolling out now, they're valued in huge amounts. It's incredible, the gold rush. And we anticipated this many years ago and started many projects in my lab that really read very much onto this idea and pushed it forward in different ways.

This is a project that we did-- actually, we presented it at the Internet of Things conference in Tokyo in 2010-- which is looking at personal comfort control. So right now, even with the wonderful nest thermostats and things like that, you're controlling still temperature on a wall. They may be a bit smarter about whether you're in the room or not, but it's really setting the temperature according to what you see on the wall.

But you really don't want to do that necessarily. You want to be comfortable. You want to have the HVAC system, the heating air conditioning system, adjust its performance to really maximize your comfort and minimize energy, the two constraints. So that wants to have a first person perspective which very much is expressed in a wearable system.

So we built a project around this. Again, we did it before 2010. And it involved this wearable. This is done by my student Mark Feldmeier, now a post-doc in my group. And you can physically tell the system if you're hot or cold with buttons. So you can give it your comfort.

It's wireless. It talks to a low power network infrastructure. And it measures constantly, at least every minute, temperature and humidity. So you're knowing what the conditions are right at your wrist as you're moving around. The wrist, of course, is open, exposed to the air.

It's measuring light levels, and also integrated motion. And now we have wristbands that all do that for athletics. Back then, this was somewhat of a power-hungry solution. So we developed our own analog accelerometer system to do it.

We essentially took a little Piezoelectric cantilever they use on disk drives to see if disk drive is dropping. It's a very, very cheap, tiny chip. And did all of the processing, all the signal processing, in analog, which at that time was much, much lower power than an equivalent active accelerometer that you would buy.

That's all different now. You can buy a micro-power MEMS accelerometer that takes about the same power. And indeed, wristbands are driving that whole market at this point in time. But here, we did it all in analog. We would integrate continuously the amount of motion you were making. And every minute, just update the sum.

And of course, we had infrastructure sensors as well. We looked at melding the wearable with the sensors in the environment. So we had sensors in the room. We had sensors

outdoors. And we basically integrated all that data together, as you do in internet of things, to try to optimally adjust the HVAC system.

Of course, we took over our old HVAC system with a retrofit. Mark put a little motor on the HVAC damper. This is an air flow and generic sensor package up there, too.

We could also open and close a window if it was more favorable, because we had a temperature sensor outside. If it was getting too windy, it would automatically close the window. It had an air flow sensor there, too.

So we ran it over a summer back then. I think it was the summer of 2010. And it worked quite well. This is before we turned the system on here, at the left. And we're using a lot of chilled air. Once we started using our system, we used much less chilled air. We estimate about 25% less energy all in all.

And even better yet, people were more comfortable. We had about a dozen people in the study. And before we ran the test over the same temperature conditions, people for the most part weren't all that comfortable in their offices. I think most of us are in that category.

Once we ran this, we were much more comfortable. People felt better. So less energy, more comfort. It seems like an approach like this has a lot of potential.

And of course, everybody asked the question, what if you have two people sharing an office? And there are a few things you could do. You could give priority to one person, maybe. But we're democratic. We don't want to do that.

Everybody had the same vote. We tried to minimize the aggregate discomfort. This is, again, taking the distance from your comfort position and trying to minimize that distance.

So here on the bottom is someone who likes it warmer in their office and turns off the chilled air. You can see that on the bottom plot. When someone comes in that wants it cooler, turns the chilled air on, and it's bringing their comfort back down toward zero, which is optimal.

That person leaves for any length of time, it turns the chilled air off, makes it warmer. So it's trying to dynamically balance between these people. And of course, it's mobile. You go to different offices, they have the system. It will automatically start cooling or heating to your preferences.

So this shows how the system actually works. You're seeing plot of temperature against humidity. It's what we call an enthalpy space. And the red crosses are where the user labeled it as being too hot. The blue stars are where the user labeled it as being too cold.

And you can see this line is a good decision boundary between these two states. And depending on what side you're on here, cools on the top. Down on the bottom, it will heat or not cool. And everybody has their own version of this. It's trained by your own data, and it can adapt and change. As your preferences change, it can actually begin to adapt what it's doing.

But this is at the heart of what it's about. Can also use other information in the switch curve, such as how much you're moving, what the temperature was in the room you were in before, which can affect your preferences, and things like that as well.

My group is also working a lot with lighting control. And for us, this is very exciting because we do a lot of work on human computer interface at the Media Lab. And that's the thing that's desperately needed with lighting controls now, in that we can, through network lighting-- which is becoming ubiquitous-- access any lighting fixture or group of lighting fixtures in a modern building. So we have tremendously agile fine control over lighting.

But we have no good way of controlling the lights. This is an example of a lighting panel in a modern building. It happens to be the panel in our new building now. This is what we live with. And to be honest with you, I don't know how to turn on the lights. I think the button in the bottom turns them off.

These are sliders for some groups, I think. But it's totally confusing. It's almost like a fuse box metaphor, where you have some presets you arbitrarily make and you write some note up here. So it's a totally antiquated protocol with a very modern and robust and agile network of lights.

So the world is waiting for a better way to control lights, and we've got several projects in my group that look at that. This is one of the first ones we did, already years ago. It's about looking at how much light comes from individual fixtures on a sensor. So the sensor could be worn. It could be just something that's in a little puck that you carry around or have on a table.

Because every light is modulated differently, you can see how much light comes from individual fixtures. You can separate the light contributions out. And then you just put a slider on top of here that controls how much light you want.

So you put this little puck anywhere, and you just adjust your lights. And it solves a linear program in real time to determine how much current should go to which fixture to give you the optimal lighting wherever you want. So that's your light switch anywhere. You bring it back here.

We've also looked at first person perspectives. If you have a wearable sensor, that's looking at reflected light off of things in your environment. So as you're moving your head around, as you're moving physically around, it can adapt the lighting dynamically to optimize the conditions as to where you're looking.

So my PhD student Nan has been wearing sensors and wearing a camera now for years in the course of her research. At this point, we don't necessarily have to build the wearable camera ourselves. We can just use something like Google Glass. And indeed, we've used Glass to control lights dynamically in work she just finished last summer. It's the topic of our last BuildSys paper. She won best presentation for that, actually.

In this work, we didn't necessarily control the lights directly. We controlled them through a model that we developed ourselves. So here, we try to think about what the optimal axes of lighting control are. You could have sliders on each one of the lights somehow, but that's not intuitive. People don't like that, as I illustrated before.

You want just maybe two axes that can get you all the lighting in the room in a natural way. Very easy to control. So what we did is light the room in many different ways. We have a very agile set of adjustable LED lights there that Philips have donated to us for this project.

And we had a lighting designer design a bunch of scenes for the room that lighting designers would do. And we had people rate those scenes on multiple axes. We had about 20 different axes.

Is it cold? Is it warm? Is it good for concentrating? Is it not? So on and so forth. And then we found the principal axes through that system. We just did an axis reductions mathematical technique.

And we used the top two axes for our control. So we just build an xy controller around the most popular two axes that came out of basically diagonal-izing this user study that we did. So in this video example here, we control the lights in the room with these two axes.

One is called focus, which is better for working. The other one is called casual, which is more for social events. And they're controlling not just color temperature and intensity, but also the physical distribution of lights. They're all together in these axes.

So now she's driving it with Glass. Why control it if you can drive it with context? So the wearable camera, the sensors in Glass, are driving her position in her two axes. That's what Glass is determining with her algorithms.

As she moves, the amount of light going to Glass changes. So it adjusts the lights dynamically to keep it at a certain level so she can see. But now it detects that she's being social because she's on the phone, she's talking. She's not working. So it moves smoothly into casual.

And then she eventually finishes her conversation. She'll go back to work. And it's getting more into focus. And of course, it's totally dynamic.

She has a friend come by. That's Gershon, playing some music and talking. And it's going back to casual again. So we don't need the light switch, necessarily, if the context can drive the lighting. And we're living again in this world where sensors project up information, context projects down. And that basically controls our environments.

And this is just the tip of the iceberg of what we can do in environments that are going to be common in the next few years.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 6

As I mentioned at the top of my talk, wearable computing and wearable sensing has been fundamental to work at the Media Lab over the last, probably, three decades now. And wearables, as you've seen throughout my talk, have been a part of the vernacular. They're part of almost everything we do now. I don't consider wearables so much to be a separate area of research or a separate endeavor. For the most part, it's a big part of what we do in all of our projects.

So, I'm going to talk through a few examples that we're doing in my group with wearables that really do leverage the wearable sensor in an IoT framework. And of course, wearables are a phase we're passing through. Eventually, I think it's going to be implantable, as far as I can see-- the next wave of computing. But that won't be covered in my talk.

As I mentioned, at the Media Lab, we had the pioneers of wearable computing with us, many of them, during the mid-'90s. And in this picture, you see a bunch of the students with their wearable rigs, out in front of the lab. During those days, they had a whole group. They called themselves cyborgs, and other names, during that time.

But they really pioneered a lot of the work we see rolling out now and have gone on to be prime movers in the area of wearable computing. This is Thad Starner over there, one of the main developers of Google Glass, actually living in that world with his private eye back in those days, and his network computer already-- again, mid-'90s. If Thad was hanging around, I could ask him any question, anywhere, and he could just look it up on the web, the way we do on our phone. But he'd do it in his wearable, in the mid-'90s.

And this is Steve Mann, a true visionary of wearable computing, doing wearable augmented reality already back then-- sensors all over his body, cameras streaming data back, all kinds of innovative visuals going on in his display. A real pioneer, and continuing to pioneer work of this sort at University of Toronto as a professor for many years now.

This is the first project I did my group in wearable sensing. Back in 1997, I did a shoe for a dancer that had 17 different sensors on it, streaming live from each foot, to make a real-time performance. So, as the dancer danced, the music would be created directly from real-time data coming back from all these sensors.

And back when I would show this work at conferences, people would laugh and think it was crazy to put so many sensors on a shoe. But we're living in that era now, so sensors are on our wrist. They have been on our foot. They will be again. This is not so different anymore.

Matter of fact, this is one of the inspirations for the Nike+. They were sponsors. They hung around with us a lot and worked with us during this time. And we had no iPod back then, so this is something that had to evolve years later.

So, this is a dancer that we worked with that was at Cornell at the time. His name was Byron Suber. And he's wearing the shoes, shortly after we built the second generation. This is the beginning of 1998. And the sensor data is streaming off of his feet, making that music in real time, with a computer and a stack of synthesizers, back in those days, that he's hearing.

So, pressure sensors on his feet are making sounds. If he lifts his foot up, we can detect that. If he bends his foot, we detect that. And we can map that right away into different kinds of musical events. If he spins, we detect with a rate gyro. So, we can launch another kind of musical event.

And we even had sonars on his shoe, so we could track his location. Here, we can see if he's getting closer to the corner. So, we can fade up another sound. And we also had electric field sensors in the room as well. And there he's stepping on an electric field sensor. And as he's moving his foot around, we know he's on top of that. We know the height of his foot. And that can be mapped to different kinds of sound.

This is a recent project that we've worked on, a few years back, that brings together all of the different building control systems that I talked about, in one wearable device. We call it the WristQue. And, of course, the wrist is coming back as a place for digital technology. Many of us wear FitBit bands or digital watches. It's beginning to catch on.

But I'm interested in not so much the wrist as a display, or a peripheral to the phone, or doing one narrow application-- but really as a source of control and interface. Because the wrist is something you can point with. You can point with your hand. If you sense where the wrist is, you can start to point. And in this device, we actually have a full inertial measurement system. We have an ultra wide band radio as part of it that can track it to within some centimeters indoors.

So, we know where you are. We know the angle of your hand in the room, if we back out the magnetic field distortion. So, you can start to see what you're pointing at. You just extrapolate the vector from your hand, from your wrist, out to whatever it intersects. You can point at things and gesture in different ways.

We also have the temperature-humidity sensors and the low-power accelerometer so you can do the smart HVAC application with this. As you move around, your comfort control can move with you. We also have a light sensor that can detect different modulations in the lighting system. So, that can be your light switch anywhere. You just put out your wrist and expose it, and control it with a slider.

There are many videos done of this by people that have visited the Media Lab. There's one link down here that shows an actual demo with the system. The wrist is potentially the gateway to the fingers. And as I mentioned, I think we're going to be touching things less and less as context mediates most of our interaction with these smart environments that will be everywhere.

But still, there will be times when we want to use a more direct interface. And the fingers aren't going to go away. We have so much sensory motor control there. They're still going to be leveraged. But I think the interface of the future is going to be really with your hands by your side. We've done a lot of work, both in the Media Lab and other institutions, tracking hands in front of you with cameras. And some great work has been done that way. But this is, I think, for a lot of things, a little bit awkward. If I'm outdoors in the crowded subway, where people are using their phones, for example, I'm just going to have my hand by my side, and just kind of move my fingers casually like this.

So, we built a bunch of systems to get at finger movement and finger gesture from the wrist. This is the first one we did, way back in 2010, using off-the-shelf hardware. The idea was to have rings. You can see it here in this conceptual diagram.

The rings can track the position of the fingers as they move. They can look at the clicks with the thumb. And you power them just from an RFID reader at the wrist, just as an Internet of Things RFID. These are just sensor tags, basically.

This is using the Intel WISP. One of our alums, Josh Smith at University of Washington, produced a whole series of these passively powered microprocessor radio sensor boards. And we power them up with this RFID antenna. And they have an accelerometer. They have a little read switch for the magnet on the thumb. It does all of the things that we claim here.

Of course, it's not optimized. We're not using a frequency that sees easily through the hand. That could be fixed pretty easily. These are off-the-shelf devices that are not optimized for the particular operations we're doing. Now, with a better accelerometer, much lower power, more suitable processor, we could make these things take orders of magnitude less power. And we can make them a lot smaller, so they become rings. And you power it up through the wrist. So, this is very doable.

My students, though, don't like having anything on the fingers. Even though some of them wear rings on their fingers, up here, they don't want to have an interface that requires that. Most of them don't wear watches, for that matter. So, they want to take everything off the fingers. And going from the wrist-- they'll wear something for an interface-- try to back the fingers out.

So, we have a few projects that looked at this. This is one done by one of my alums, David Way, where he's got a 3D sensing camera that looks at finger motion in 3D. And he runs machine learning on it. So, he can have a set of gestures he can get with his fingers to control any number of remote functions, including texting. Although I don't think we're going to be texting in the future. We're going to be something more leveraging context. But he did it with texting.

So, here's a live example of David's interface. It's recognizing the gestures as he's lifting his fingers. It's recognizing pitches. It's quite robust. And as these depth-sensing cameras get integrated into everything, including mobile devices, one of the real goals of industry

now, the power will come down. They'll get smaller. You can start to put it into a wristband. So, here, he's texting. He makes a mistake. He can backspace and fix it.

This is another project we have, trying to get at finger gesture through the wrist, by my student Artem Dementyev. And here, he makes a pressure-imaging wristband that images the tendons as they deform when you pinch your fingers. So, it's an array of pressure sensors, very low cost. You could even print this onto a band. And very low power. And he does machine learning to identify certain states. So, it is quite effective.

This is a video that he did with our WIST paper. So, you can see the tendons move. And he images it with this. Again, quite low power-- you can put it in a watch. And this is after a quick training. It recognizes basic pinch gestures, does it quite well. It knows, also, if you're holding things. You can train it for that.

Of course, he can wake it up at very low power with a specific gesture. So, at just bare microamperes, it can recognize that and wake up. He can control things in his environment, like lights. It's straightforward.

My favorite application, he controls his bike lights automatically. Because we have to squeeze when we pull our brake. Recognize that trivially, and he can flash his warning lights.

This is another wearable project in my group. It's again a project with Artem, in collaboration with Cindy Kao from Chris Schmandt's group. And it's a fake fingernail that is an actual touchscreen. So we built a Bluetooth touchscreen complete into a fake fingernail, with battery, transmitter, and touch controller.

So, this is all the layers, here on the right, of our system. You can see it working. Battery lasts several hours of actual use. So, here's detecting the swipes. Detects presses. And here, he can actually draw with it. You just look at the centroid.

So, here we have a simple interface, lasts certainly at least a day of use-- even more than that, because you're not using it all the time. And now, your fingernail is part of your interface.

My students who have been doing the lighting work are also very interested in wearables. And Nan Zhao wanted to go beyond lighting in rooms, to think about lighting your personal space, changing your appearance completely with wearable lighting. And, of course, there are health effects of lighting. Illuminating yourself with certain wavelengths can make you more alert, awake, or really change your mood in different ways. So, Nan thought about trying to bring lighting onto the person, onto the face, in a very intimate way. So, Nan has developed this system she calls Halo, which is a ring of lights around your head. You control it with a Bluetooth app.

So, there's Gershon, controlling the way he looks through a simple interface. And he can express himself in very different ways. Of course, that lends itself to a certain social

dialogue. Here's my other student, Juliana, just taking a walk through campus with her own Halo. And she sees Gershon there. So, they notice each other. And their Halos can start to express their inner feelings.

She won an award from the wearable computing conference just some months ago for this work. The large Halo, of course, is somewhat ungainly. Great proof of concept. But they've gone further and made a more compact version, which you would wear like a set of eyeglasses, but still illuminates the face quite effectively. So, this is one of my alums, Nan-Wei, demonstrating this system running in real time, just illuminating her face in different ways. So, this might be a slightly more practical Halo, if you want to call it that.

Going further, we have a post doc visiting my group now, Katia Vega, who is running what she calls beauty technology. She's very well known for this work for years. And that's taking the actual interface and the sensors, and putting them right on the skin with makeup. So, she makes functional makeup that can work as an interface right on the skin itself.

So, this is going beyond wearables, before we have implantables. It's just physically treating the skin with different kinds of compounds, none of which are toxic, and using that as an interface to whatever you want. So, this is an example from one of Katia's shows, where she has her models wear different kinds of sensors.

So, in this case, there's a switch on the eyebrows. And of course, she puts the LEDs physically in with the rest of the makeup. That's strain gauge on the eyebrows. And then she can put a switch on the lips as well. So, again, the interface is applied to the body in this case. The sensors are applied to the body, just as you make makeup.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 7

So tools are a fundamental part of human experience. They enable people to do many more things than they can do just with their body. A skilled artisan sees a tool as an extension of their body. And they just adapt to it in different ways.

But what if the tool could adapt, as well, to what you're doing with the tool? The tools of the future of IoT will be smart. They'll have sensors on them. They'll have intelligence. They'll be connected. They'll know something about what they're meant to do. So they can collaborate with the user. You essentially will have human-robot cooperation with the robot in your hand.

So we've done a suite of projects to explore this area. Actually, my alum, Amit Zoran, is one of the real pioneers of this field. And he started this work with something he called The FreeD, which is a milling machine that's precisely tracked. And it has lots of robotic affordances. It can pull your hand back if you go too close if you violate a CAD constraint. It can physically pull your hand back.

It can also deflect the bit, or it can turn the bit off. So it can decide on its own to inhibit what you want to do. But you have judgment. In the future, robots will make everything, perhaps. But there's a period now where human judgment is still unique. And the judgment of a skilled artisan or artist can say, no, I want to penetrate this CAD standard, I want to do something you're not letting me do. So you push a button, and it lets you do that. It records that. And it can adapt.

So Amit did a project to explore this. It's a very pioneering project, called The FreeD. Amit was inspired through travels through Africa over many years, seeing people make many of the same sculpture. This is the device pulling the hand back. There's a fiducial here at left. And if he goes too far, it physically will pull the hand back. In this case, it actually turns the motor off. But you can push the button and penetrate the constraint and keep on milling.

So here, he's milling out a saber-toothed tiger. It's one of the first projects like this Amit had done. He's not a skilled miller by any means. But with the aid of his tool, he's able to reveal the sculpture that's inside of the block. And it still has his characteristic. You see the striations, the stripes, the style. That's Amit's hand physically on the tool.

Again, as I mentioned, you can decide you want to violate the CAD constraint. So he thought of a few ways of handling that. In this case, he's milling out a giraffe. And if he's penetrating the giraffe's neck, the CAD model will adapt. In this case, it can fit a spline, or it can use a different, more closely-fitting model, instead. So he's got a few different ways of addressing this.

It's intriguing. He used many different subjects in his work. If you look at their physical art, they would do in the real world. This was an illustrator. This was a person who made

drawings. This is a jewelry maker that would sand parts over jewelry. This guy did 3D printing. This was a violin maker.

Their cat that they made with this tool kind of have some of the characteristics of their art. They didn't plan this. This is just what happened. So even with this digital tool, something of the personality of the artist carries over.

And then he analyzed the work patterns. Because you're tracking people with this tool all the time, you can see people that don't have so much skill with tools kind of all over the map here and different kinds of working styles. This guy here is a violin maker. So he's used to milling wood, anyway.

And this carries over into this digital tool, because he's using, essentially-- he explores a bit here-- he's using, more or less, one style of work. Here, he takes a break. He stops and just rests. And then he goes back to work again and then touches up. And he's done before everybody else.

We've also made airbrushes. So it attracts the airbrush. And it knows where you are in 3D space and figures out what color to spray. You can override if you want. So we have lots of different pictures we made at the lab with that. This is a paper here we published on it.

My student, Pragun, has also made a non-conformal printer. So this is a regular printer. It's got an ink jet on it. It's also got a tracking and digitization tip. So it is tracked, as well, with the same tracker we used on all the other devices. So it knows its position in 3D space and knows its angle.

So you basically can digitize an object by running this tip over it in different ways. It does a fit to smooth out the data from that tip. And then you can go with a print head and print on it. And it will just turn the print head on where it's supposed to do. So you can raster it. It doesn't have to be flat. It could be totally non-conformal. And you can print as you desire.

So you can now start to print on something that is not at all flat that you define at the time of the tool just by moving the tool across it so it knows what the object is and fits to it.

He improved it quite a bit before he finished his thesis. So this is the final result he had. And you could even, again, defeat what the CAD model wanted you to do and print something on your own. It would capture that. And he studied all of those things.

He also did what we call BoardLab. This is an electronics tool. A lot of us grew up doing electronics and still do electronics. And even now, if you're trying to probe a circuit, you have to look at the waveform on the scope or the logic analyzer. You have to look at the schematic , the PCB board layout. It may be a data sheet. This is a way to bring it all into your hand.

So it tracks the position of the tip. It knows what point you're probing. It can show, certainly, the waveform you're measuring, show the reference voltage-- because it accesses a database-- show the part in the schematic where you're touching it, or the PC board layout, even bring up a data sheet if you desire.

So it's a seamless way to interact with information when you're doing electronic tests.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 8

The last thing I'm going to talk about are what we call sensate materials. So in this world that we're coming to, we see Internet of Things physically embedded into materials. You have materials that have sensing capabilities that can be distributed throughout the material in a natural way.

And my metaphor inspiration for this is human skin, right? So we have receptors all over our skin. The nerves are already processing and synapsing the data as the data flows through to the brain. So you have something that's a scalable, sensate medium that is robust but also captures and reduces and transfers a tremendous amount of data.

So can we make materials that are like this? Make what I call a sensor processor "soup" inspired by electronic skin. And again, the global term that we like to use is "sensate media." So you have processing and sensing distributed throughout a material and densely networked.

So this is a research area that goes all the way from discrete circuits that are embedded in flexible or malleable media all the way up to substrates that are printed or even fabricated using fab techniques, or the silicon fibers and things like this. It's a very active area of research in many, many communities, including MIT.

But in my group, we've played with these ideas for years now. There's always been a project running somewhere in the group that pokes at this in different ways. This paper here in BT Systems Journal talks about some of the basic concepts, references a lot of the work as of the beginning of mid- 2000s. It's moved tremendously over the years.

This is one of the projects we did. These are all little sensor nodes that are on a piece of flex, and they're interconnected. They process locally all their data, lots of different sensors. And you plug into the edge, and you basically receive the data. Here's another one we did with rigid circuit boards and flexible interconnects of different sorts, again measuring lots of things, producing the data in the network as it flows through. And again, something like a sensate skin.

We've been printing lots of things over the years. And this has certainly got a very broad future as we can start to print more and more sensors and more and more electronics. This is a sensate floor where we printed all kinds of different sensors with a little network and sensor concentrator devices we would embed on it.

This is a guitar, with the actual skin of the guitar physically printed on a printer, and the whole thing is a capacitive sensor. So it's a decorative, aesthetic object, the way most guitarists appreciate. They like their guitar to look cool. But anywhere you touch this guitar, it will respond. So the guitar itself is a sensitive surface continually, as well as being an aesthetic one.

This is a current project in my group. We call it sensor tape. So we physically take a strip of what is basically tape. It's a flexible electronic circuit. We put down circuitry, at least in those days. At first, we would physically put circuits down that would measure different things and certain network functions.

But now we basically have it all on a continuous piece of flex. So we can take many meters of this. It's cheap to fabricate in bulk using flexible circuitry techniques. And it's measuring lots of parameters along its length and just sending it down on a network to the end. You plug in, and it can respond.

So this is Artem's video that describes sensor tapes. Again, it was inspired by tape. It looks like tape. It measures lots of things. In our case, we physically have inertial measurement units. So we can measure the tilt and the twist, so the sensor tape knows its conformation.

It has proximity sensors, so you can measure how far away you are from it. You can capture shapes in different ways. And Artem figured he'd try to improve his posture with it. So this is one of the demos that he did. It's physically evaluating his dynamic posture.

So again, we see it as a utility. You can make this material, use it for lots of applications. It can be a tool. It can be embedded into a fabric, into a product. And of course, going further, you can look at different twists to this. This is something that my student Jie Qi has been doing together with Bunnie Huang, one of the alums of the Media Lab. It's sensor stickers. So this is a bunch of stickers, just like children would have to play with with their sticker kits. And my daughters love playing with sticker books still.

But here we have electronic components that are on stickers. So you can make your circuit physically with copper tape or some sort of conductive stuff you stick on and take your sticker and just put it down. So Jie has a whole following of people. Schools actually all over the world have been using Jie's stickers. You can buy them on Amazon, other places. And schools have been using them as part of their curricula to teach kids to engage with electronics in a very creative way.

So here is an example where somebody's made a flower. That's the process of-- there are LEDs and sensors responds to different kinds of things, lots of examples. She's made a book recently as part of her PhD work, where the book is beautiful. She's got LEDs. She's got circuits she makes with copper tape. And they interact and evolve and change in different ways, depending on how their programmed and what they're sensing.

In the future, you'd be able to print a lot of this. Because again, as I mentioned earlier, you can start to print more and more electronics, more and more sensors. It's still difficult to print high-quality sensors for lots of things. Different kinds of sensors need different kinds of processes. But a tremendous amount of work is happening to try to bring this all together.

And so we've poked a little bit at this and begun to explore printed sensor tape. We can print different kinds of sensors. This is discussed In Nan-Wei's paper. And I think there's a rich future to this. I mean, you can think of printing a low-cost sensor that's passive. It could be powered up by stray RF signals or RFID coming from a phone.

You can put it on the back of a wall to measure moisture. For instance, my roof is leaking now. I want to know if this is bleed-through from the paint or if there's really water up there. I could just use a piece of the sensor tape on the wallboard. It could measure that, and I could interrogate it with my phone or whatever. It would tell me if it's really wet up there or make an alarm.

This is an example of this done by a visitor in my group for a few years now. He's been a professor at the University of Tokyo, Yoshi Kawahara. And it's a printed sensor that he put together to look at hydration on the ground and hydration of a leaf. He printed it all with an ink-jet printer, and he printed an antenna to harvest energy from the Tokyo TV tower.

So you can put this in the ground and charge up over time with energy coming from the TV tower and then give wireless updates every so many minutes of the soil hydration, the hydration of the plant. So again, this points to what this future wants to be.

Beyond IoT – Ubiquitous Sensing and Human Experience (Joe Paradiso): Video 9

So this is a conclusion more I guess as a brief epilogue attesting to the way the world is changing very quickly. I've given you lots of little examples of projects we're doing, a lot of technical vignettes pointing at how things will change in the near future as sensors become ubiquitous and the way we plug into them becomes more and more intimate in the future.

And, again, we're just seeing the tip of this. The sensors are getting out there everywhere. They're piggybacking on the back of devices that are already in place. Sensors get cheaper, you start seeing them in quantity in almost everything now.

And once we get protocols that can share the affordances across the different devices, we're living in a different world very quickly. It's what I would call a phase transition because the devices are already there for the most part. There'll be more of them there but there are already a lot of them there. So once we have these protocols that can share these affordances, share these different bits of information, share contexts and sensing information from our vicinity, we're going to be living in a very, very different world.

And there are lots of opportunities, lots of challenges. That's a very exciting time to think about how to interface humans to this ubiquitous nervous system that's coming quickly.

So I've summarized at the end of this talk here a few resources that you can consult for further information on a lot of the things I've been talking about here. The website for my group is one of the prime places if you want to get more information on the particular projects that we talk about, and that's listed right here.

In addition, there are many conferences that really publish state-of-the-art research that's going on in these areas. I publish in a lot of these conferences, my colleagues do. It's a great source of information to really see what academics and researchers are doing in these areas.

The big one in ubiquitous computing is Ubicomp. There's another significant one called PerCom. Mobiqus is another example. Of course, the IoT conference needs no introduction here. There's some that are domain specific, like Pervasive Displays-- displays everywhere, how we will interact with them in a seamless way. Conferences for that.

There are more. These are just some of the ones that we've been active with that are great resources.

For human-computer interface research, which is very much going to be apart of Internet of Things, because the future interface to world of computation is going to be through, for lack of a better word, IoT or ubiquitous computing. The main one, the mothership, is CHI. It's an ACM conference. We all publish there. That's a 3,000 person conference. You'll see a lot of great state-of-the-art work there.

UIST is a bit smaller, more on systems and technology but a lot of great ideas are presented in UIST. Another one that's a little wilder, lots of wonderful speculative ideas and some really great concepts is TEI-- Tangible and Embedded Interaction.

For embedded sensing and the built environment, there are lots of conferences on smart buildings, but there's one in particular that combines building systems with sensor networks in the Internet of Things in a very good way. It's called BuildSys, and that's the URL there. We publish there sometimes.

For sensor networks, there are some great conferences as well. Sensor networks, again, are the foot soldiers or the leads, so to speak, of ubiquitous computing. They are the things that will enable it that will be everywhere. And the flagship conference for sensor networks is called Sensys. Lots of great technology and applications there.

IPSN, Information Processing and Sensor Networks is another great one. The European Workshop on Sensor Networks and SenseApp. This is about applications of sensor networks. All great, cutting edge research from the community here.

For wearable computing, there are a bunch of conferences. The original one that we started actually here at MIT in the late '90s is called ISWC. I think it's the International Symposium on Wearable Computers. Still going on. I think it's part of the Ubiquitous Computing conference now but it has a separate identity.

BSN, Body Sensor Networks. You'd begin to get into medical applications but also human-computer interfaces and Internet of Things. BodyNets is a similar conference. The IEEE EMB Committee is very involved in wearables and medical monitoring. It's a great, high-level technical conference.

And then other conferences like BioDevices. There are a bunch of them springing up. Again, looking at wearables and the IoT infrastructure. Mainly for health, although there's some of it about interface.

For sensor technology, IEEE Sensors is one of the big flagship technologies. Anything new about sensors, including the journal and the conference, are great resources. PowerMEMS-- was just here in Cambridge, actually, last month-- is one of the main or the main conference now for energy harvesting.

The way you power these sensors is perhaps to harvest energy off of any available source, and this is a great conference for frontier breakthroughs in that. IEEE RFID, of course, very close to Internet of Things.

So there are also a lot of magazines and periodicals that are relevant for this field. The most interesting magazine for Internet of Things in my opinion right now is IEEE Pervasive Computing.

I've been on the editorial board there for a number of years and I've helped shepherd articles through it and really have enjoyed the process. It really exposes lots of great research in ubiquitous and pervasive computing and now more and more Internet of Things. The URL is here and lots of great articles and a good resource. Very easily readable articles for the most part too.

And then there are journals too that are in between a magazine and a hardcore technical journal, and probably one of the most relevant ones for us in ubiquitous computing is the Springer Journal of Personal and Ubiquitous Computing. So you find lots of articles there that cover this field from variety of viewpoints.

So thank you very much for staying through this lecture and completing this course, and I wish us all a great fortune in this future world that comes quickly of IoT.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health (Dina Katabi): Video 1

Hello, everyone. I'm Dina Katabi. I'm a professor at MIT. I work on wireless systems and wireless networks. And today, I'm going to tell you in this class about emerging wireless technologies that you can use for the internet of things-- in particular, emerging technologies with applications for wireless positioning and for smart homes and smart towns.

So let's start with the first topic-- indoor localization. Now, I'm sure that you have used GPS repeatedly for navigating outdoor spaces and going around, finding your way outdoors. That's great. But unfortunately, GPS does not work indoors. And everyone who works in the industry and also academia realizes that the next frontier is indoor localization.

Why are we interested in indoor navigation and indoor localization? So there are so many applications that can be enabled by that. Let's just think about a few of these-- for example, indoor navigation in spaces like malls or museums. Imagine entering a mall, for example, and immediately on your cell phone, you can say where you want to go, which store that you want to go to, and then it will allow you to navigate the space just by looking at the screen where you are in the mall.

Now, look at this from the mall's perspective and the retail stores and all of those guys. So they are interested in knowing how people navigate their spaces so that, for example, if you are there looking at the rack of shoes, they can send you a coupon for a discount on shoes. But they need to localize you. If you are looking at hats, they don't want to send you the coupon for shoes. They want to send you a coupon for a hat.

And another application is inventory. So we can revolutionize inventory if we know exactly-- imagine knowing every box, where it is in the warehouse, when it entered the warehouse, how it got moved inside the warehouse, when it exited. It just changes so many things that we do today.

So how can we do indoor localization? So for GPS, we use a wireless signal. So the natural thing is to think, OK, why don't we use RF or wireless signals for that, too? And this is exactly what we do in our work. We use RF signals for indoor localization.

But it's not that easy. Now, the key challenge when you do this is something called multipath effect. So let me explain what that means. The traditional approach for trying to localize a device based on wireless signal follows one of two approaches. The first one is that you can think that you have your transmitter, and you can use the power of the signal. So if you have your receiver away from the transmitter, you can imagine that the power would decrease as the receiver gets away and away from the transmitter.

Now, the second traditional approach is to look at the spatial direction of the signal. So if the signal is coming from this direction, I know that the source is in this direction. Sounds reasonable, isn't it? Unfortunately, it doesn't work in indoor scenarios. So let's see why.

All of the signal bounces off all the objects in the environment. It bounces off the furniture, the ceiling, the floor, the walls, everything, and also our bodies. So when the signal bounces off all of these things, the signal is not just going to go from the transmitter to the receiver. Everything is going to bounce around before it reaches the receiver. And that bouncing effect is what we call multipath.

So let's see this visually. So here, you see this blue dot, which is our source. And when it transmits, due to the obstacle in the environment, the signal is going to follow the two paths that you see. So if you look at this, of course, the angle where the signal comes from is not the direction of the source. In fact, there are multiple angles, and none of them is the direction of the source in this particular case. And the power [? of sorts, ?] the received power, is not reflective of the distance. Because the received power depends on how these two waves are going to combine together at the receiver, whether to cancel each other or to emphasize each other. So we cannot use it as a distance metric. So multipath is going to affect how we think about localization in indoor spaces.

Now, what we did is to try to step back and say, OK, so let's turn the table around. Why can't we use multipath as a signature of the environment to actually enable localization? So let me tell you what I mean by this. So basically, when I have my transmitter-- say that it is here, the transmitter signal-- the signal is going to bounce all over the place. But that bouncing pattern is going to be characteristic of this particular location.

Now, if I take that transmitter and move it here, again, when you transmit the signal, it bounces around, but that bouncing pattern is going to be slightly different, because I moved my transmitter. And it's going to be characterizing this particular location. And if I move it even further, then the pattern will change even more. And therefore, we can use that bouncing pattern, which is a multipath profile, as a signature of location.

So let's again see visually an animation of that. So now we have three sources-- the blue, the green, and the red. So the signal from the blue guy travels these two paths. The green guy, because it's very close to the blue guy, the two paths are almost the same as the blue path. Now, the red guy is close, but not as close as the green guy to the blue guy, so we can see that the paths are more different.

So now let me show you something we call the multipath profile. So what I'm going to plot here is the power of the signal on the y-axis as a function of the spatial angle-- so power from this angle, then from this angle, from this angle, and so on. So for the blue guy, we see that there are two peaks, basically. The first peak corresponds to the first path of 35 degrees, and the second one corresponds to the second path at 140 degrees.

Now, as you would expect, if we go to the green source, you will see more or less the same peak, slightly shifted. Now let me show you the red guy. So now, as you see for the

red guy, the signal still looks kind of the same, but it's more different. It's shifted more in comparison to the green guy.

This means that we can use a multipath effect as a distance metric. So we can have a few reference sources in the environment, and then we can compare everyone else to those reference sources to know the new devices and their distance from the reference sources using comparison of multipath profiles. And this is what we're going to do.

Now, there is one thing that's still missing here. How do we get these multipath profiles? So there is a standard way in wireless systems to get these multipath profiles, which is to use an antenna array. Each one of these antennas, when the signal falls on them, they receive that signal. So each one of them will receive that signal, y_1 up to y_n , and actually, it's a textbook equation to generate the multipath profile from the signal received and this antenna array.

So it's very easy to generate the multipath profile if you have an antenna array. But the problem is that antenna arrays are bulky and big and expensive. There are so many antennas there. So can we generate the same thing, but with a single antenna?

So here's what you can do. You can take your antenna and slide it on the body of the device. So imagine your access point, and the antenna is just sliding on the access point. And now, as it's sliding, it's actually tracing the location of so many antenna elements. And now you got an antenna array, virtual antenna array, and you can apply exactly your same equations-- the textbook equation that I told you about-- and you generate this multipath profile.

OK. So now we are equipped to check how this algorithm behaves in practical scenarios.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health
(Dina Katabi): Video 2

We now have an algorithm and I'm going to see whether it works.

So in principle, everything that we talked about is not specific to any technology. You can apply it to Wi-Fi signals, you can apply it to cellular signals, you can apply it to something called RFIDs. And I'm going to show you the application of a wireless technology that is called RFID.

So RFIDs are stickers. They are just a passive device like what you see on the slide. And you can take an RFID and stick it, for example, on my cell phone. And then when you shine a wireless signal in the environment, that RFID will reflect the signal, and it will modulate it by its own ID.

So let's say the ID 15. So you shine a wireless signal, and then the reflected signal will come back and say, I am RFID 15, I am RFID 15.

So if you can localize RFIDs, then you can localize all objects in the environment. Because you can put these stickers on everything around. And you can see now the location of these objects. And if they move, you can discover how they are moving.

Of course, the other thing that is really nice about RFIDs is they are very, very cheap. So, a few cents. So you can buy so many of them and tag everything in your home, everything in your space. If you can apply this localization algorithm then we can know how things are moving around.

We can apply this to a particular application that people care about a lot, which is elimination of customer check out lines. So imagine in a store, putting RFIDs on the objects in the store, putting a few RFIDs on the basket of the customer. And then if we can localize the objects into the basket, we don't need these long customer check out lines.

The objective is to immediately, as a customer takes an object off the shelf and puts it in her basket, localize the object into the basket. So let's see whether that is possible.

So I'm showing you, in this video, an experiment that is done based on RFIDs and the algorithm that I described to you. So here's the customer, and here are the RFIDs. You can see them on the basket on the shelf, and also on the red Pringles, which is what this customer is interested in. Here is the multipass profile of all three objects, or all three RFIDs.

And as you can see, the red Pringles, which is the red multipass profile, are much closer to the black pattern coming from the RFID on the shelf. So the algorithm immediately can tell that the red Pringles are still on the shelf. So now let's see what happens when she takes the object off the shelf and puts it in her basket.

Here is what happened. Of course, when the red Pringles moved from the shelf to the basket, their multipass profile has changed. So we see the new red multipass profile, and now it's actually very close to the orange graph, which is in the RFID on the basket. So the algorithm immediately knows that this object got and moved to the basket.

And so now we have an algorithm that can see how objects are moving around, and can localize them immediately to the basket of the customer if she picks an object. And of course, if she puts it back on the shelf, it would know that the object is back on the shelf.

OK. So now we talked about this technology in the context of RFIDs. Do you think that it works for Wi-Fi? Everything that we talked about is about signal. So in fact, the protocol or what characterizes Wi-Fi versus RFID is not here at all. So it should, in principle, work for Wi-Fi.

And of course, there are few tweaks that you can find in the literature that we have. And I have references at the end that you can look at. But let me show you a video of applying the same kind of algorithm to Wi-Fi.

So this is a video of two robots. So one we call the delivery robot. It has a Wi-Fi card. It has, actually, a laptop on top of a Rumba, and that laptop has a Wi-Fi card. And the other laptop, which is on the Rumba, is the target.

So the delivery robot it's supposed to know where the target is without anyone telling it anything. So basically, there is no camera that this guy is feeding off. There are no additional sensors. He's just using Wi-Fi signals that are emitted by the target, to localize that target and find where it is.

So now we see the robot is moving toward the target. Again, based purely on RF signal. And to make it harder, we made the target run away and try to escape being found. And now you can see the delivery robot actually discover that it had to go around the file cabinet to find the target.

So one thing to keep in mind is that when I describe the algorithm to you, I will say we slide the antenna on the body of the device. In this case, we don't actually need to slide the antenna, because the robot is moving. So we can leverage the natural motion of the robot to slide antenna naturally.

And with this, I showed you an algorithm and a design for localizing using RF signal. I showed you that it worked with a variety of technology. Of course, I encourage you to look at the references at the end of these slides. And I'll look at additional details that complement this algorithm and make it possible to implement it in practice with the Wi-Fi and RFID example that I showed you.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health (Dina Katabi): Video 3

So now, we're going to move to the topic of smart homes, and how new wireless technologies can enable smart homes that can adapt to our habits and log them, and also adapt the environment to our comfort.

So wouldn't you like your smart home to be able to know, for example, when you wake up in the morning, so that it can open the shades? It would know when you enter the living room and sit on the couch, so that would turn the TV on, and would turn to your favorite channel.

Maybe when you go to the shower, it would know that, after that, you want your coffee. And then it will turn the coffee machine. So by the time you are ready for the coffee, your coffee is ready. So all of that is great, and that's the vision of the future. How can we get there?

We need our smart home to be able to do that, to know something about us, how we are moving. What are we doing, where are we, and what is our pattern of movement? And, of course, the way to do that is using wireless signals. So what we do is we use something called device-free localization.

So in the previous section, I told you about localization. But I was always assuming that there is a device on the localized object, whether it's a sticker that you put on the cell phone, or whether it's a Wi-Fi transmitter that is embedded in your cell phone. But there is some device that is always on the object that you want to localize, and, effectively, you are localizing that wireless device.

But if you want this to be seen less in the home, then you want to localize a person without any device on hand. When I wake up, I don't have wireless devices on me. We can do this, and that would lead us to this new, very new and emerging technology called device-free localization.

So we're going to use the reflections as the wireless signal, as they reflect off the body of the person, and analyze, also, reflections to detect how the person is moving. So, again, it's a very, very high level. So you see on the slide the wireless signal.

So we know wireless signals can traverse walls. So even if the device is in a different room than the person, that's no problem. And then, also, they reflect off our body. And some of these very minute reflections will come back to our device. And if our device has smart algorithm, then it can analyze these minute reflections and discover how we are moving in the space. So before telling you in more details how this works, I'm going to show you what is achievable in using these kind of technologies today.

So in this video, you see that the device, actually, is not in this room. The device is in the office adjacent to this office. So the device is behind the wall.

So you see an office. You see a person standing, and the device, as we said, is in the other office and is shining the wireless signal, is analyzing the reflection of the wireless signal off the person's body.

So now, let me play this video for you. So notice the red dot on the side of the screen. So that side of the screen actually shows the screen of the device. The red dot is where the device thinks this person is now. And now, notice how this red dot is tracking this person very accurately.

And remember, this person doesn't have any wireless device. No cell phone, no [? other ?] ID, nothing on himself. He might as well not know that he has been tracked by some device in another room. And you can see, it's very accurate.

Not only we can track the full-body motion, but we can track body parts. So, for examples, this person can point his hand at the lamp, and the device will know where he pointed his hand and will turn the lamp off, or can turn the screen off by pointing at it.

And next time you leave the room and you forget to turn the lights off, you don't have to go back. You can just stand there and point at those lights, and can localize the way you are pointing and turn off the lamps.

Let's see other applications in the context of the smart home. So we already talked about one application, which is gesture control of appliances in your home by pointing at them, but there are many others that you can imagine. So, for example, gaming. So today, to play a video game, you have to stand in front of the gaming console.

Because if you try to, say, dodge a bullet by hiding behind the couch like the guy in the picture, for example, then your gaming console just loses you. It cannot find you, because it cannot see you.

But with this, you can augment these video games with the ability to interact with the furniture. And in the home, you can even still play the game as you go between rooms, because the signal goes through walls. So you can have much richer games.

Not only this, we also, now, we talk a lot about smart cooling and heating. So your home can know where you are, how many people there are in a particular room, and can adapt the cooling and heating and save you energy, and, therefore, save you money, also.

Another application is in health care. So, nowadays, people live longer, so there is a large number of older people in the US and all over the place. And older people are very vulnerable to falling. So they can fall, and these falls can be fatal, in many cases, or they can cause serious injuries.

So you can do fall detections. Because I showed you that this device can localize the people in 2D, but it can imagine that the elevation is another dimension. So you can get localization, also, in 3D. So let me show you results from that.

So this result is from the same device, but this is for fall detection. So remember that red dot that was following the guy? So that was in 2D, and now we are looking at the elevation of the red dot. And in this experiment, we see three activities. One guy moving, the other guy is sitting on the chair, and the third guy is emulating a fall.

And you see elevation as a function of time. And you can clearly see that, when the guy falls, the elevation goes to the floor level, and that is detectable. And that is different from sitting on a chair, because, in that case, the elevation does not go to the floor level.

OK, so there are many applications. I'm sure that you can think of others, like home security, for example. Just being in your bedroom, and you can see if somebody, for example, jumped from the window and entered your home, and you can take the right action, hopefully, for that. So how does it work? So we talked about it at a very high level, but let's now dig into the details.

We said that we are going to use wireless reflection of the body. So we transmit a signal. It can traverse walls, and if reflects off the person's body. And we are going to analyze these minute reflections to see how people are moving in the space. And now, of course, the wall is just to show you the harder example. Of course, if the percent is in front of the wall, that still works.

So how can we use these reflections to detect the distance or the location of the person? Now, the very simple principle that we all know is that distance is equal to time multiplied by speed. So the distance of this person is equal to the signal reflection time multiplied by the speed of light, because all the signal travel at the speed of light. We all know the speed of light, so we can detect the distance by measuring the reflection time. But how do we measure reflection time of such signals?

Now, the first thing that might come to your mind is, why don't I just send a pulse, a wireless pulse, and wait for that pulse to reflect back to me, and then I can measure the time between sending the pulse and the pulse arrival. While that works in principle, in practice, it's very costly and inefficient to do it that way.

Because, if you want to measure times of reflections, and you need electronics that work very, very fast, on the order of picoseconds, and that would be very expensive. So we need a different way for measuring the reflection time. So what can we do?

So we're going to use a principle called FMCW, Frequency Modulated Carrier Wave. So let me explain how it works. In FMCW, what you have is a linear relation between the frequency and the time. So say this is your transmitter signal. Its frequency changes linearly with time.

Now, if you transmit the signal, and the signal went and reflected back off some object and came back to you, then the reflection looks like this, another line, but separated by some reflection time. And our objective is to measure that reflection time.

But another thing about FMCW is that, at any point in time, the difference in frequency is related to the reflection time. Because the signal is linear, as we see from the graph, and that relation is reflection time is equal to that change in frequency divided by the slope of the line.

So we know the slope, because we chose that slope. We can choose a slope that we want. And if we can measure ΔF , the change in frequency between the transmitted and the reflected signal at any point in time, then we get the reflection time. But then, how do you measure ΔF ?

So it turns out that actually measuring ΔF is not that hard. There is a very simple device that is in almost every single wireless device, like in my Wi-Fi and cellular systems. It's called the mixer.

So the mixer is a device that takes two signals and gives you as the output a signal that has a frequency that is the difference between the two frequencies. And if you think about it, for people who know more about Wi-Fi, for example, we say that Wi-Fi is transmitted at 2.4 gigahertz.

But the Wi-Fi bandwidth is 20 megahertz, so which means that, when the signal is transmitted, the signal is up-converted, and then, when it's received, is down-converted. So basically, to down-convert the signal, you are really subtracting, in some sense, the carrier frequency to go back to that 20 megahertz of bandwidths.

So we're going to take a mixer, and we're going to put into the input of the mixer the transmitted signal and the received signal. And then, as the output, we're going to get a signal that has the frequency difference between the two.

So it has a frequency ΔF . Now, we are going to use our friend FFT, the Fourier transform, and we get something that spikes at ΔF . OK, cool. So now, we can measure ΔF . From ΔF , we can get the reflection time. And from that, we can multiply by the speed of light and get the distance.

**Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health
(Dina Katabi): Video 4**

So, are we done? Of course, it's not that simple. There's more to be able to capture these minute signals and identify how people are moving in the space. In fact, the stuff that we just talked about in this section on localization comes back to haunt us again, which is multipath. If you are a wireless engineer, you would know that multipath is the biggest problem, the phenomenon that we always talk about when you talk about wireless signals indoors.

So, because of this multipath effect, let me remind you. What is multipath? Multipath means that you have so many reflections bouncing off all objects in the environment. So, effectively, it's not that the signal just reflects off the person's body. The signal reflects off the table, off the chair, off the wall, off everything else.

So, as a result, you don't just get one spike in that delta F as we simplified in previous slides. You get many, many spikes that reflect all the objects in the environment and their reflections. And because the wall and the furniture and all of that stuff reflects more signal than the person, in fact, the reflection from the person will be just completely immersed and lost in these bigger spikes of multipath effect. So what can we do?

So, we need smart algorithms that can disentangle the reflection of the person's body from all that multipath effect. We're going to use a simple principle. So we know that static objects don't move. And therefore, if you look at the reflection of static objects, they don't change over short time scales, while the reflection of a moving object is going to change. The assumption that this underlying here is that most movement in the environment is associated with humans, either moving himself or moving some object by pushing it or carrying it.

Then, we can localize a human by looking at changes in these reflections and ignore all the static reflection. So we take the signal at some time, t . And then we take the signal at a later time, say, t plus 30 milliseconds. We subtract these two. And now, all the reflections from the static objects have not changed, so they disappear, while the reflections of moving objects remain. And now we can look at them much more clearly, as we can see on the slide.

Now, if you look at my slides, you see two peaks from my example that I just showed you. And in this particular example, there was only one person moving. So, how come there are two peaks, and those two peaks are separated by two meters? So it can't be that the person is just like, in two places at the same time. So what's going on here? So, again, I told you that multipath is really the issue for any wireless engineer and when people look at RF signals indoors.

So, it turns out that there is not just a static multipath. There is something we call dynamic multipath. So, basically, the signal reflects off the person's body. And, after reflecting off the person's body, it can reflect off some object in the environment. You get

these composite reflections which contain both the reflection of the person's body and some other object in the environment, or maybe even more than one object in the environment. And those reflections change with the moving person, because he's a component in these dynamic reflections. So they're not going to disappear. And this is why we got these two reflections-- one from the table one, one from the person-- in this simplified example.

So how can we distinguish the reflection from the person from the dynamic multipath? One way to think about it, which is actually not an effective way, is to think about, oh, the reflection from the person is going to be higher power. It's not necessarily true, because we are dealing with really very minute reflections here, and that is very susceptible to errors. And also, you don't really know what the person might be-- the indirect reflection might traverse less obstacles or less attenuated obstacles than the direct reflection, for example, if you have a wall in front of the person.

So the better option is to consider that the direct reflection from the person is a direct path. So it's a shorter path, by definition, while the indirect reflection traveled more distance. And therefore, you can just look at that and say, OK, so the reflection that came directly from the person has to be faster, and therefore it's smaller delay than the reflection from other objects. And therefore, this is the first reflection in this case.

And this is the way we can eliminate the dynamic multipath and zoom in to focus on the moving person. So, this just tells us the distance from the device to the person and back to the device. Now, that doesn't tell us the location of the person. So, to get the location, we need to do a bit more.

Let us just look at what we have. So we transmit the signal to the person, got reflection, and got the signal back. We measured that distance, the round trip distance, and let's call it d . So, if we know that the person is d from the transmit to the receive receive antenna, then, effectively, it means that person could be anywhere on an ellipse where the four sides are the two antennas.

Now, know where exactly on that ellipse the person is, we don't know. But we can answer that question, but just have another received antenna. And now, if we have another received antenna here, then we can measure a second distance from the transmit antenna to the other receive antenna, which is d' . And now we have another ellipse. And the person has to be at the intersection of the two ellipses. And if we use directional antenna, then, the person has to be within the beam of the directional antenna so we can detect which one of the two intersection points the person is located at, which of these two intersection points, and eliminate the one that is behind the antennas.

So far I explained this within a plane, like a 2D environment. Of course, we live in 3D. So, in 3D, the extension is fairly simple. Instead of talking about ellipses, you would be talking about ellipsoids. But the same principles still apply.

So, with this, we end the section about looking at emerging wireless technology for making your home smart. We sat down. Your home can track you. And, based on that tracking, it can discover what you are doing-- like, when you wake up, you leave the bed. How you are moving is a home. And then, based on that, can make the home automate or take certain actions to make your life more comfortable. Or you can use it, also, as we said, to point at appliances as a gesture-based control. So the next thing that we're going to talk about is smart health.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health
(Dina Katabi): Video 5

So now we are going to talk about emerging wireless technologies that make your home smarter in the sense of taking care of your health.

So we simply have been talking a lot about smart sensors in the home that measure temperature, measure air quality, and can adapt the home to our comfort. But then you can start asking a different question, why can't my home measure my breathing and heart rate, that are my vital signs?

Now, you might wonder, why would I want my home to measure my vital signs? So if my home measures my breathing and heart rate, then I can ask my home questions like, do my breathing and heart rate reflect a healthy lifestyle? I can ask my home, does my baby breathes properly when she sleeps? I can even ask my smart home about my elderly parents, OK, does my elderly parent have arrhythmia at night, for example.

Not only that, some of you who got the chance to know about sleep apnea-- and if you ever try to get a sleep apnea pass and went to a sleep lab and they ask you to sleep with all these sensors on your face and your chest, then they tell you, sleep normally-- which is virtually impossible. So with this technology, if the home, if the environment can measure our breathing and heart rate without having to put all these sensors on our body, then you can just get the sleep apnea test at home why you are in the comfort of your bedroom, sleeping normally. If our home knows this information about us, then it can adapt the lighting and the music to our mood.

But how do we measure breathing and heart rates? Some common techniques for measuring breathing and heart rates can be really cumbersome. So particularly for breathing, if you want to measure the breathing of a person, you either have to put in a nasal probe on a band, or you have to put a chest band, which is very cumbersome, particularly if you wanted to have ubiquitous measurements. And if you want to add heartbeats, then you either have to make the person wear a pulse oximeter, or you can make them wear a wristband, which is not as accurate as the pulse oximeter.

But if you want both, then you need all of these sensors on yourself. And if you are one of the people who are fine with all of these wearables, that's OK. But at least we know that there is a segment of the population that does not like wearables, and is uncomfortable with wearables. These are the elderly, and the kids. Try to put wearables on your kids, and they take them off, they throw them away. It's a non-starting point.

So what we want is to have the environment do these measurements for us, without having to change anything about our pattern of living, without having to wear new sensors on ourselves, or having to charge the sensors, to remember to take them off, charging them, putting them back. So how can we do this?

In fact, what we are saying here is that we want the environment to monitor breathing and heart rate from a distance, without any body contact. Now, you will be sitting like there, and imagine somebody, you are sitting there, and they don't touch you, they don't put anything on your body, but they are actually now measure your breathing and your heart rate.

So what I'm going to tell you about is a device that does exactly that. It can measure your breathing and heartbeat without body contact from a distance. And it's a technology called Vital-Radio that can do exactly this, and it does it accurately. In fact, it is comparable to FDA approved devices for measuring breathing and heart rate that you have to put on your body.

Not only this, this device can monitor multiple users at the same time. And because as we said earlier, wireless signals can traverse walls, this device can monitor a person that is not in the same room. So you can have one device, and monitor people in multiple rooms at the same time.

So how can we do this? The basic idea is we are going to use wireless reflection of the person's body. This is an idea that we just introduced in the previous section. Now we are using it to measure breathing and heartbeat.

Now, we know that wireless signals gets reflected off the person's body. So if you transmit a signal like this, the signal will reflect off the person's body, will come back to the device. And now if we can measure the distance of these reflections, then we can measure the change in distance during each of the inhale/exhale motions, because basically your body, when you are breathing, your chest is inhaling and then exhaling. And as a result, that distance is changing between you and the device.

Now, I just told you about a mechanism for measuring the distance, when we were talking about the person walking around, and we are able to measure his distance, and we are multiplying the reflection time with the speed of light. So that mechanism is very good at measuring the absolute value of the distance, but here we want to measure distances up to a millimeter.

So we are not talking about finding the location of a person where you want to localize him, to within half a foot. We are talking about millimeter changes in distances. So we need to augment that [INAUDIBLE] and by something that can measure much smaller distances. And I'm going to introduce a new idea, which is we are going to use the wave itself.

Wireless signals are waves, so any wave has a phase, as it propagates. And that phase changes with distance. And so there is the relation like what I have on my slide, which is that the phase is related to the distance by this equation. So if I can measure changes in the phase, I can relate them to changes in the distance.

And this is exactly what we are going to do. We are going to measure the changes in the phase of the reflected signal. From that, we are going to see changes in the distance. And from those variations in the distance, we have got the inhale/exhale motion of the person.

So this is how we are going to measure the breathing and heart rate. Now, you might wonder how do we measure the heartbeat?

So it turned out that when the blood is ejected from your heart, it exercises a force on the rest of your body. And that force causes very minute motion in your body. So your head actually jitters as a result of that force that is coming from pushing the blood. And that kind of jitter, both in the head and actually the vibrations of the skin allows us to measure the pulse, which allows us to measure your heartbeat.

So I'm going to show you these results. So in this video, you see a person sitting and reading. And what we are doing here is I'm showing you his inhale/exhale motion. So this is his breathing.

So now he's holding his breath, and you see that motion-- actually, it's now steady. It's not going up and down anymore, because he has held his breath. He is going to hold his breath for about 30 seconds. Don't worry about him, he is alive and doing very well. So now he's breathing again.

And I'm going to play this video again for you at five times higher speed. So now look at his chest motion, and you see when he holds his breath, that it's very much synced with the signal staying steady.

So let's remain on the signal. So this is very much the same as the signal I showed you into the video. These are the inhales. These are the exhales. And these blips on the signal, actually they are his heartbeats. So now we've got the breathing of the person, and his heartbeats.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health (Dina Katabi): Video 6

Now we've got the basic idea of how we are going to use wireless signals to get breathing and heartbeat. But, I described to you the system with a person who is static-- not moving. Now, of course, when the person is going to move, that is going to affect the wireless signal. What happens, for example, as a person moves a limb, like moving a hand or moving a leg? Of course, if we don't identify these kinds of motions-- limb motions-- they can impact our estimate of the breathing and heart rate and cause errors.

Let's look at an example here. In this example, the person was breathing at the beginning, and then around the one-minute mark, you see a big limb motion. Actually the person did something like this. Now, I can see that there is a major difference between breathing and the limb motion. First, the limb motion is much bigger scale, of course. But, also, breathing is a periodic motion, while a limb motion is typically non-periodic.

What we do is to identify the periodic from the non-periodic motion. We have a periodicity test of the signal. We'll identify windows that have non-periodic motion, like limb motions, and eliminate those from our estimate. And then we'll look at the windows that have the periodic motion. And from that we apply frequency analysis to extract the breathing and the heart rate.

Another issue that you could wonder about-- what happens when you have multiple people in the environment? How their breathing and heartbeats are going to affect the wireless signal and therefore interact with each other. For example, here, we see Alice and Bob in the same room. And the signal reflection off the body of Bob is going to come back to the device. Also, the signal that's reflected off Alice's body is going to come back to the device. And these signals are going to collide together at the device.

Actually, not just Alice and Bob. We talked about-- OK, signals reflect off all the objects in the environment-- tables, chairs, walls, everything. Also, the signal reflecting off a table is going to come back and affect the device itself. And all of these signals are going to collide together. And when signals collide together, then the phase of the signal that is received is a combination of all of these phases. So it's a mess, basically. Everything that we talked about so far, about using these phases to detect these minute motion, is not going to work unless we have some mechanism to disentangle these reflections.

How do we disentangle these reflections? We're going to use what we have learned so far, which is positioning using wireless signals. We learned that wireless signals allow us to position objects based on these reflections off their body. We can use that as a filter to separate the reflection from a different part of the space into their own bucket, like you see on the slide. And then, once we do that, we can zoom in on the signal reflected from each one of these bucket in space separately.

And now, actually, we can apply the algorithm that I told you about, which is to look at the phases. Because now, the phase of the signal reflected off Bob's body is going to be

separate from the phase of the signal reflected off Alice's body. Now, of course, there's going to be some reflection off static objects, like the table, and the chairs, and all of that stuff, and those reflections don't show any periodic motion. So you just toss them away.

There is an underlying assumption here that the people are not very, very close. Of course, if two people are very close, then they're going to fall in the same bucket, and you would think-- or the algorithm would think that this is one person. Let's put it together. What we said is that we have three steps for this algorithm. The first step, we transmit a wireless signal and you observe its reflection off the person's body. The second step is to isolate reflections from different part of the space into separate buckets, so that you can analyze them separately, using filters that are based on wireless positioning. And the third and final step is that, once you separate the reflection based on space, now you can zoom in on each bucket which refers to some area in the environment, and then analyze the phase of the signal to detect breathing and heartbeat.

Now let's see how the thing works in practice. I'm going to show you a video that shows results for tracking two people. And, actually, in this video, we are tracking the breathing from behind the wall. So, in this video, we see a person sitting there, and the device is tracking his breathing. There is a second person that entered the space. And the minute the person sits down, then the device detects him, and starts tracking his breathing as well. And then it's tracking both guys. In this case, the device is in a different room, behind the wall.

Now I'm going to show you more details of the evaluation of this device, called Vital-Radio. So, we compare it to a baseline, which is-- as I said-- is an FDA-approved device called Philips Alice PDX. And, we run a large number of measurements with both the baseline and our device. You can see the baseline in the picture. That device requires you to wear something on your body, so there is a chest strap and there is a pulse oximeter that you have to put on the person's body to take the measurement of the baseline. In contrast, Vital-Radio is just-- you see in the picture the antenna of the device. It is a distance from the person.

And these experiments are done for 14 different subjects, males and females. We took a total of 200 million measurements. Then this total number of experiments lasted for a few minutes, but the total number of experiments were over a period of a whole month. We can take that many measurements because we are using wireless signals, so each measurement takes just a few milliseconds in principle.

So now, let me show you the results. I'm going to show you the accuracy of measuring breathing and heart rate as a function of the orientation of the person. You can see on the slide that we experimented with four different orientations, the person facing the device, the device being in the back of the person, and then left or right. Here are the results. You can see on the y-axis, the accuracy, on the x-axis is the orientation. And if we show you first the breathing results, in comparison, as I said, to the FDA-approved device. And, as you can see, it's pretty accurate. The accuracy that's detected in the breathing is about

between 97% to 99% of the FDA-approved device. And, here is the heart rate. And, again, we see a very high accuracy, around 97%, 98%.

Now I'm going to show you a video that is particularly close to my heart. It's a video of a baby monitor. Here you see the baby sleeping. This is actually the night vision of a baby monitor. And, you can see that it's a video. Time is passing. Here you can see the time is passing, where I put the arrow. So now we augmented this baby monitor with a breath monitor from Vital-Radio, and a heartbeat monitor. Now you can see the breathing of the baby, the inhale, exhale, and the heartbeat. The heartbeat of the baby is about 127 beats per minute, which is perfectly fine for a baby his age.

So, all parents are typically very afraid about their baby having SIDS , or they want to make sure that the baby is breathing properly while they are sleeping. Next time, don't poke your baby. You can just use Vital-Radio to detect their breathing without having to touch them or bother them.

Wireless Technologies for Indoor Localization, Smart Homes, and Smart Health
(Dina Katabi): Video 7

So, with this week, we reach the end of this class, and I have presented to you emerging wireless technologies for the Internet of Things. We saw technologies for wireless positioning indoors, we saw technologies for the smart home that allows your home to be able to track your activities and adapt to them. And finally, we saw that we can measure breathing and heart rate of people without having devices on their body. And that information could be used for adapting the environment to our needs.

And I have a few references at the end of the slides for you so that you can find the papers where these results that I showed you are published. And you can dig deeper into the algorithms and the system implementation. Thank you.

Smart Cities (Carlo Ratti): Video 1

Hello everybody. My name is Carlo Ratti. I'm professor at MIT, where I run the Senseable City Lab. And what we'll do today is actually look at the city as a Cyber-Physical System. To start it, let me begin with one example.

If you look at this, that's Formula One racing. And something interesting happen in Formula One racing over the past 10, 15 years. 10, 15 years ago, in order to win a race, you need a good physical infrastructure, primarily a good car-- a good driver as well.

Well, today, in order to win a race, you still need that, but you also need a Cyber-Physical System. You also need the system which is made of thousands and thousands of sensors onto the car, collecting information in real time, sending information to those computers where it's analyzed, it's processed, and decisions are made in real time.

This is what we will call a Real Time Control System. So a system made with two components, a sensing component and an actuating component. And sensing and actuating is even the principal of many dynamic systems or many living systems. When we see each other, when we meet each other, we sense each other, we collect information from each other, and then we respond to that information.

Now the amazing thing is that our cities are today starting to behave a little bit like that Formula One racing car. Our cities have been covered, layered with many different types of digital information, of sensors, of networks. And this is creating a new condition-- it's an IoT condition, it's an Internet of Things condition. It's about the digital and physical world converging. It's about bits and atoms coming together with us in the middle.

And again, let me give you one example. Think about the cell phone network. Well, if you surprising today to think there was a time not that long ago when humanity couldn't live without mobile connections. Actually, the first commercial cell phone network was developed by NTT in Japan in 1979. And they took around 20 years to get to one billion connections.

Now, you might ask, we got 7 billion people on the planet, how much more could this grow? Well, according to Ericsson, if we never miss the prediction in this space, we would be at 50 billion in 2020. How could this be? Are we going to have 17 smartphones in our pockets. Well, not really.

We started working with networks in order to connect with people with other people. And then we move at connecting people with data, people to machine to data, and more and more, machines to machines, M2M. Things talking to other things almost as if every atom out there in the city were becoming both a sensor and an actuator. And this is really one way to characterize what is happening today with IoT, Internet of Things.

Smart Cities (Carlo Ratti): Video 2

All of this sensing in the city is producing an incredible amount of data. This is what we often refer to as big data. And if you look at this, which is an estimate by Eric Smith when he was at Google-- if you take all the data produced by humanity until 2003, well, arguably, that is more or less the same amount of data we now produce every 24 to 48 hours. The quality of the data is not exactly the same. In the first chunk of data, you had Shakespeare and Moliere and the Bible and everything else. You know, we don't have a Shakespeare every weekend. But the quantity, the sheer quantity of data, is very similar.

There is another definition of big data that I like. It's by a colleague and friend at University College in London, Professor Mike Batty. His definition is, big data is what you cannot put in an Excel spreadsheet. And if you think about it, it's actually quite profound, because when you got big data, then you also need new tools. You cannot use the traditional tools to analyze it, to visualize it. But we actually need to develop new tools in order to really understand what is going on. And this data can be actually classified and put into different categories. And this is what we call in research, opportunistic. It's data we collect by analyzing existing networks. Think about the cellphone network. It produces an incredible amount of data, and this is the data we can use to better understand how the city works.

Then there's other types of data. There is user-generated data. It's all what we generate every day just by going online, by posting a picture on the internet, by posting something else, by tweeting. So, all of this is what we generate all together every day, and again that can tell us a lot about the city and how it works.

And then the third category is when we actually develop sensors and we put sensors in the city in order to go and see what is going on in one part of the city or in another part of the city, and that's really what is purposely sensed.

Smart Cities (Carlo Ratti): Video 3

So let's start with the first type of data, opportunistic data. Look at this. What you see here is actually the city of Lisbon. It's a beautiful visualization done by Pedro Cruz, who was a researcher in our lab who looked at billions of billions of data points taken from GPSes mounted on taxis. It really allow us to see a city like we couldn't see just a few decades ago, a city like a living organism.

If you look at the next visualization, it's also something else we pioneered in the lab. It's about using cell phone data in order to better understand what happens in our cities. And it's an old project we're going to show you. It's a project back from-- dates back from 2006.

And what we did in this project was actually take a lot of-- anonymizing the aggregated cell phone information in the city of Rome in order to better understand what happens. And while we were collecting data in Rome, at the time something that this had not been done, and then we were showing the information at the Venice Biennale, a big exhibition about art and architecture that happens every couple of years in Venice. And that was in the summer of 2006.

And what I'll show you now in this little video is actually something quite interesting that happened that summer. Italy and France actually were selected for the final of the soccer World Cup. It's a little bit like the European version of the Super Bowl.

What you see here is the city of Rome. You see the Colosseum, the river. It's the morning, people moving here and there with their cell phones. And then it's the early afternoon.

In a moment, the match begins. Silence. Nobody talks anymore. France scores, Italy scores.

Halftime, people make a quick call. And then end of normal time. The first overtime, the second overtime. A famous headbutt by Zidane, player. And finally, Italy wins.

You see that night everybody goes to the center to celebrate. The following morning, again, people go to the city center to meet the winning team and the prime minister. And then by the end of the day, everybody goes to a place called Circo Massimo, where, since Roman times, people go and celebrate. And you see a peak in the lower part of the screen in a moment.

So really, we can use all this data from the networks in order to better understand cities or entire countries. What you see here is actually all of Great Britain. And what we did was build one of the largest ever made social network, known internet social networks, based on how we all communicate with each other and use this information to better understand regions over space.

Every time we actually have a-- make a call we create a link between two places in space. And then if we aggregate all of these links across the whole country, we get something we could call a human network that captures really how we're all connected with each other. And then we take the human network and we try to analyze it in order to better understand communities over space. And this is what you see here, how the algorithm is partitioning space.

And what you find, you find an interesting geography. It's not based on the past, not on history and politics. It's actually based on the natural way we all connect and communicate with each other.

And if you look at it carefully, you also see that some regions are very close to the ones we know, such as Scotland. But others, such as Wales, are actually very different. Wales is divided into three parts, with the central part of Wales being very connected to the center of England. And for those of you who are interested in the mathematics behind, you can look at details of the paper that looked at this, which is really based on network analysis, in order to better understand partitions, not of the network, but of space itself.

And of course you know here, you can do it in other contexts. You can find it online, different results. Here is actually all of the United States. And there's something similar across all of the states, and see how boundaries actually change compared to the boundaries we know and we are familiar with.

Of course, all of this was about the most typical type of data, of opportunistic data-- data collected by networks, communication networks-- but there's many other type of data that we can use in order to better understand how a city works. And what you see here is the data from credit cards. You see the country of Spain, and you see how people are using credit cards and how this changes over space and over time. Actually, the time is quite a specific time. It is the time which is called Semana Santa, the Holy Week, just before Easter.

And so you see how data's happening day by day and you see the changes across the country in what people buy and the transactions that people do with credit cards. Again, when you mine this data, when you analyze it-- here we just see visualize, but when you analyze it, you can get very, very interesting information. For instance, we get information about how different cities compare, as you see here, how different neighborhoods compare, almost like a dashboard of the city and its economic performance.

In terms of opportunistic data, we've also done a few experiments in-- on our own campus here at MIT. And so just to do a little step back, what I want to share with you here is actually about the idea of a campus or a city as people looked at it in the 20th century, the idea that a city would be something where you divide in a very precise way different function-- a place for living, place for working, place for leisure, and of the connection between them. And then this idea of mixing functions that have been very popular since post-World War II, but that today is becoming something even different, a

new type of relationship and distinction between public and private space that really changes the inner structure of the places we inhabit.

And what you see here is our beautiful campus at MIT. You see Boston. You see all of MIT, almost like a little city inside the city. Harvard up there-- don't bother.

And MIT was one of the first places to be totally covered by Wi-Fi, in the early 2000s. And that has been creating a very big difference in the way people work. What you see here is the way people used to work. Students in particular used to work in those computer rooms, as you see here to the left, and how we can work today.

Now this is a bit extreme. I looked for the most depressing computer room I could find-- no daylight, all artificial. And what you see to the right is very beautiful, but also something that is not like that during the winter, when it can be very, very cold. But this idea of the changing workplace that you see at the top and at the bottom of this image is something very, very present.

Well, I'm telling you all of this because again, we can use opportunistic data to better understand what is going on. In a certain sense, some of these changes have been produced by networks, by the Wi-Fi network, by wireless connectivity. So if you go and monitor the network itself, you can probably get a good understanding of what is happening on campus.

And what you see here, you can see all of the access points on the MIT campus. You can see all the Wi-Fi nodes. And then you can monitor activity over time.

So you see here a typical day. People wake up in the morning in dorms, and they move to the center of campus, and they move around. So really basically how occupancy changes over the 24 hours.

If you look at total occupancy-- you see it here on this slide-- when you look at the top, that's total activity on the MIT campus measured by looking at the Wi-Fi network. And what you see, you see people getting to campus around 9:00 AM. And then continuing to the 9:00 to 5:00 PM, you see this kind of peak at 5:00 PM. Quite a few people leave, but many people keep working till late, till 8:00, 9:00, 10:00, 11:00 PM. And still in the middle of the night you see a lot of activity.

To the left, it's a Monday. And the same pattern you find on Tuesday, on Wednesday, on Thursday. Not on Friday. Like all over the world, activity slips away on campus quite quickly on Friday afternoon.

But then you go to Saturday and Sunday, they're almost like normal day. You just removed the 9:00 to 5:00 people, and you still see a lot of activity happening. Something quite interesting-- you see a little dent at the end of Sunday around 9:00 PM, and that's when we all say oh my god, tomorrow is Monday again, and we realize all the things we haven't finished.

Well, if you look at this, which is the overall activity, but if you actually look at this space by space, room by room on campus, you get many of these signatures, many of these curves, that tell you how occupancy changes over time in different points of campus. And then if you analyze this-- and you can do a Fourier transform, you can do other type of analytics-- then what you find is really looking at how different parts of the spaces we inhabit work different-- again, by taking all of this data that the network produces. So in a certain sense, the network is changing our life, but the network becomes an opportunistic way to sense some of these changes and to better quantify what is going on.

And we've seen all of this on the MIT campus. Of course, you can apply it to other contexts. What you see here is actually the same type of analysis applied to the Louvre Museum in Paris, one of the greatest museums in the world. And again, this information, collected in an opportunistic way from the networks, can help us better understand human occupancy and ultimately design better buildings, museums, or entire cities.

Smart Cities (Carlo Ratti): Video 4

So let's look now at the second type of data, and that's data that's user-generated. It's data that we produce every day, just by taking a picture, uploading it online, by tweeting, and many of the other activities we do on social networks and elsewhere. In the lab, we were one of the first teams to actually start looking at his wealth of data in order to understand how it could tell us something about the city.

For instance, look at this, at all of the pictures we take, and we put them on Flickr. Then, we can analyze them to better understand something. In this case, for instance, we worked with the city of Florence to look at how tourists move. What you see to the left, you see tourists all over Tuscany, where they go and take pictures, see some of the main hot spots.

And then you can zoom in to the city of Florence, which is the bright spot in the middle. You can zoom in and look at different parts of the city. You can zoom in all the way, because you've got GPS coordinates all the way down to the single monuments. What you see to the right is actually the Duomo and the places where people are taking pictures of the Duomo.

Now, if you want to do something slightly more sophisticated, what you do is that you add a little script to look in every picture, the time stamp-- when it was taken. And if you do that, then you can build trajectories. You see that people have taken pictures in different places. And you have this time stamp, so you can build a trajectory.

So what you see here, for instance, is Italians and Americans as they move in Tuscany. And look at how different it is. To the left, you've got Americans. And, of course, if you come from the United States, you go to Tuscany, you might go once in a lifetime or once every 10 years. You will go to the main places-- to Pisa, to Genoa, to Florence, and so on.

If you were Italian, you might have been there many times. You know those cities very, very well. So look at how different it is, the way you go and try to look for little hidden gems, little places in the middle of the countryside that are outside the main tourist attractions.

And the same type of analysis you can do in other ways. This is actually something that our students were quite excited about, was about finding the best place to go partying in Barcelona. Again, you look at all the pictures on Flickr. You analyze them, and you can get that kind of geography based on what people do in the city. I believe they even managed to get a scientific paper approved looking at the correlation and proving a significant correlation between Britons and parties in the Catalan capital.

And if you look more broadly at all of Spain, for instance, you can use this information in a slightly more sophisticated way. And you can look also at other interesting dynamics. For instance, the big issue in Spain is drought. The country gets drier and drier as you move from the winter months to the summer months. And then what we did here was

actually look at the pictures, analyze the amount of green they have, and we found a very simplified way to monitor drought as it progresses through the country.

So, instead of using a traditional way to monitor drought, which is putting a number of sensors or measurement stations across the country-- you need to maintain them, they're quite expensive-- is a way to really do this almost by borrowing the eyes of all the people who are there. We called the project-- "The World's Eyes"-- "Los Ojos del Mundo." It really is a way to monitor together things in our cities, in our landscapes, and use as sensors our own cameras.

Well, all of this is just to show you about how we can use pictures. Of course, there's much more online content we can mine. Tweets are a great wealth of data we can use today. We can download it and analyze it. For instance, by analyzing tweets, we discovered something interesting-- the fact that there is a significant correlation between the length of tweets and the excitement we have about something we are tweeting about.

What you see here, we looked at 40 million tweets for the 2012 golf tournament. In the video, you will see every dot is a tweet. In the lower part is short tweets. In the upper part is long tweets. And you see the average is the yellow line in the middle.

And what you see here, you see how tweets densify when there is an event and how the length of tweets changes. And mathematically, you can see here that it follows a very, very predictable pattern. Again, you see the densification, the excitement on the system, and how the average changes. So, in other terms, how we can use all of this data to better understand how our cities work, but also how we work as humans. This can tell us something about society and about our behavior.

Smart Cities (Carlo Ratti): Video 5

So we looked at the opportunistic data, data we collect, by operating, say, a cell phone network, but we can use it to better understand the city. We looked at also user-generated content, but in many cases, we really want to sense data [INAUDIBLE] by deploying sensors. Today's sensors are becoming cheap, small, it's very easy to deploy them, and so we can collect more and more information in the places we live by deploying sensor networks.

In the following project what we did was look at air pollution in China. Pollution in China is a big issue. We focused in particular in Hong Kong and Shenzhen. They're basically the same metropolitan area, but they're divided by the mainland China border. And instead of monitoring air quality in the traditional way, which is, again, by deploying a bunch of measurement stations, big measurement stations, in the city, what we did, we used very, very small sensors that people could carry, measuring their own exposure to pollution as they move through the veins of the city, and sharing this information with each other. Here's a little video.

[MUSIC PLAYING]

So this project was about looking at how people themselves can carry sensors. But as we said before, sensors are becoming so inexpensive, we can use them to track many, many different flows. For instance, in this project we look at trash. If you look at that computer, today you know everything about it-- every chip in that computer you know, where it was produced, how it moves on the planet, how it was assembled to become that machine. Today we know where our objects come from. We know everything about the global supply chain, but we know very little about what happens to things when we throw them away.

Sometimes if you think about the computer when you throw it away, this is what happens to it. A lot of electronics from the United States being shipped illegally to Asia, from Europe to Africa, so our idea was if you could design a little, little tag to put on trash to start following trash? It's a little bit like when you go to hospital, they put a tracer in your blood and they follow it through your body. How could do the same thing in the scale of an entire city?

And we couldn't find anything off the shelf, so we actually worked with Qualcomm to engineer those tags that we could put on trash. What you see here is the first deployment we did in the city of Seattle. We invited volunteers to come. We had 500 volunteers come with 3,000 pieces of trash. On all of them, we put a little tag for tracking-- every possible thing about electronic e-waste, and also all the way to a banana peel.

And after tagging all of them, we started following them. So here's the 3,000 objects they day of deployment in Seattle. After a few days you see some of the main landfills next to Seattle, but an actually big surprise how far some of the staff started to travel, sometimes in unpredictable way. Look at the trace that went all the way to Chicago and then down to

Bath, California. 40 It stayed moving after 1 month or 2 months all across the United States.

So what can you do with this? If you're an engineer, if you're an architect, you can use all this information in order to optimize, call it the, removal change. So not the supply chain, but what happens to things after we throw them away. Think about how much energy could be saved if we were to run analytics on those traces and actually try to optimize the movement of trash.

The other thing that's very important is that if you share this information with people, then that can start interesting dynamics of behavioral change. The idea is that big data means that we know a lot about the environment, about the consequences of what we do, and so if this information is shared and made publicly available, then it can enact changes of behavior.

Just one example was somebody who came to us after the project. During the project we were sharing all the trace information with people, and they came to us and said, you know, I used to drink water in plastic bottles every day and then throw the bottles away and forget about them. But now after the project, I know that those bottles go a few miles from home to a landfill, they will stay there forever. So because of this, I stopped drinking water in plastic bottles. And so this is about the power of big data when you share with people, and actually changing the way we do things.

There's a final thing we discovered about this project, and that actually happened more recently. It happened when a burglar came to our lab here at the MIT campus. The poor guy stole a lot of things including tags and computers that tell you where they go. And this is what happened.

[MUSIC PLAYING]

So we really looked at the three different types of data sources we have is about data that we can get in an opportunistic way, data we can get from user-generated content, and data we get from sensors we actually deploy in cities. Now, in the next section, we will actually look at how to make sense of all this data and how to use it and combine it in the city.

Smart Cities (Carlo Ratti): Video 6

When you got so much data about the city, the important thing is then how to combine it. And that's one of the things we're looking at at our lab in Singapore. It's a project called LIVE Singapore! Really combining making sense of different data sources. What you see here is a little video about the project. Imagine a city, living city, where you know in real time all what is happening around yourself. And what you see on the platform is actually again, data from telecommunication networks captured and collected in real time.

You can see other types of data. Here is actually energy consumption in the city in temperature. You see when it gets very hot in Singapore a lot of air conditioning goes on. And then you consume more energy.

You can see what happens during special events, dates, the Formula One racing in Marina Bay Sands and everybody moving there. Also unexpected things. Here you see all the taxis, the little dots, and you see rain. You see actually when there's a lot of rain all the taxis disappear because people need to move in different ways.

And then how the city almost expands and shrinks because of congestion. So this the travel time you need to go from one part of Singapore to another part of Singapore. And then how will all of those local flows intersect with the global flows? So you see all of the containers getting into Singapore, all the planes getting into Singapore and getting out of Singapore. So how local and global flows interact. So that really becomes the beginning of a platform that has all this real time information about the city. And it can become a very powerful tool both for urban analytics, but also for us citizens to better understand what goes on around ourselves.

If you look at this, then an important part is visualizing the data, is collecting data, is combining, is fusing the data, but also it's visualizing. And really what we call in Singapore is Data Collider. It's a platform that allows us in a very simple way to combine different data sources. It really allows it in democratic way, so that everybody can easily play with the data.

Here's a little video that shows how it works.

[VIDEO PLAYBACK]

[END PLAYBACK]

And really this becomes a way to democratize data visualization to make it accessible to everybody.

[MUSIC PLAYING]

And so really the importance of combining different streams of data to find platforms to do that and to fuse the data, and also to visualize it. Because visualization is the key thing in making data access more democratic.

Smart Cities (Carlo Ratti): Video 7

What we saw so far was how to collect and use different type of data in the city. But as we saw in the beginning with the Formula One racing car example, the important thing is not only sensing but is act waiting, is how to use this information in order to transform the system. And that's really what we would look at here with one particular example, the example of transportation in the city.

Now of course, in many cases, just by sharing the data you can transform the system. The very fact of visualizing and sharing the data can allow people to make different decisions. Think about ways that looks at traffic information in cities, where the very fact that people have access to that means that then in the morning we might make different decisions in the routes we take to go from point A to point B. So information per se, when shared can enact important changes. And we also saw it briefly in some previous examples.

But then in some cases you want to build intelligent analytics on the top of data and use them to better understand the city and transforming. Let me give you one example. In this project, we looked at taxes in New York City. We use a big database of taxi information that was made publicly available by the Bloomberg administration. And what you see here, you see every dot is a pick up or a drop off. A yellow dot is a pick up, a blue dot is a drop off. What you see here is JFK airport. If you zoom out, you can see all of Manhattan and the boroughs, so all of the taxis and the patterns of getting onto a taxi or off from a taxi.

And then you can ask yourself, what if we could share a taxi? What if you could share mobility? Well, you know, these days we like to share many things. We like to share apartments, online platforms, couches on couchsurfing platforms.

So what if you could share mobility better? And if you look at this within two points, almost any points in your city, you've got hundreds and thousands of trips every year that potentially could be shared by people. Here is a little video that explains the idea.

From mathematical point of view, you could frame the problem in the following way. Imagine you want to take everybody to destination exactly when they need to be there, give or take maybe a couple of minutes, a small delta of delay, but a very limited delay. Then what would be the minimum number of taxis you need in your city to satisfy the demand of the city? Well, again, when you go dig data, sometimes you need to look at new mathematical tools to do this.

From mathematical point of view, what you want to do is ask the following question. Imagine you will take everybody to destination exactly when they need to be there-- well exactly plus or minus a little delay, so you imagine a small delta, one, two, or three minutes. Then what will be the minimum number of taxis you need in your city to satisfy the mobility demand?

And when you got a big data, sometimes you need to develop new mathematical tools to analyze the data. Just the sheer quantity, sheer size of the data, makes it, in some cases, traditional techniques fail. In this case, we develop something shareability network. We look at network science in order to turn a dynamic problem into a static one, can use this static representation of the network in order to run an optimization algorithm.

What you see here you see an example of the shareability network. You see all of these little dots that actually densify. If you increase the delta, if you can tolerate a slightly longer delay, then you see how the network is densifying. And when you analyze it, you find something quite extraordinary. You see from the graph there at the top, that in New York, you'd actually cut 40% of the infrastructure, 40% of the fleet of taxis, and still take everybody to destination exactly when they need to be there.

And incidentally, we started working on the project a couple of years ago, and the first results came to the attention of Uber, and since then, we started actually collaborating with them. And as you might know, UberPool, which was recently launched, does exactly the same-- allows people to really share trips and share mobility in the city. Now this is what we can do today, again, because of real time information, using real time information to actuate the city and transform.

If you look at some other changes, you see, again, how Internet of Things, it could transform mobility even more. What you see here for instance is what the self-driving car sees. It's a three-dimensional scan of the city that looks at the city almost like a human eye via a three-dimensional model. And that is the input for the self-driving car.

Now the interesting about the self-driving car is not that you don't need to keep your hands on the steering wheel. Of course, yes, you can-- you'll be able to shamelessly text while driving. But don't think that's not the key point. The key point, we believe, is the fact that the self-driving car changes the dynamics between public transportation and private transportation. In other terms, in the morning, the car can give you a lift when you go to your office, and they can give a lift to somebody that's in your family or to anybody else in the city.

So again, it is creating a hybrid system. In this hybrid system, if you combine it with what you said before-- before was about ride sharing, here we're talking about care sharing, using the same car, that can be used by different people over time. With the combination of both of them, we say we could have a city, such as New York, or Boston, or London, or Singapore, big city, or small city, where we take everybody to destination when they need to be there, but with 20% of the cars we have today. Now think about how different our cities would be if you were to remove 80% of the cars, which means also removing a lot of parking and reducing a lot of the cost of the mobility infrastructure we have today.

If you look a few years down the line also when there will be more self-driving cars on the road, when everything would be self-driving, then we can think about other changes. If you look at this, that's a very well-known traffic light. And traffic lights, of course, appeared when cars appeared on our roads, a little over a hundred years ago. But if you

think about an intelligence system where a mobility is self-driving, then perhaps we don't need to stop at traffic lights anymore. We can actually have flows that magically mix when we tell cars bumping into each other, something like this. Don't try it yet.

Well again, this is something quite simple, but it can become an interesting mathematical problem. What you see here is actually the intersection, and you have different flows competing for the piece of space in the middle. You need to model it. And some of the work we're doing is really looking at that, and looking at how we can design the most effective [? slot-bay ?] system and how it would compare with some of the systems we have today.

So what you see here, you see a real intersection in Singapore. The amount of cars is exactly the same to the left and to the right. But to the left, you see negotiating axes with the most intelligent traffic light we know today. And to the right, you actually see a [? slot-bay ?] system. And look at how different it is after just a few minutes. In terms of delays, you introduce them to the system, in terms of number of cars waiting, to the left and to the right.

Well, these are just some examples of how IoT, how the convergence of the physical engage the world can actually change our cities. In this case in particular, looking at transportation. And how data we send from the city can then be used for modeling what is going on. And ultimately to transform the environment we live in.

Smart Cities (Carlo Ratti): Video 8

We've seen how a lot of IoT can have an impact on our cities, for instance in terms of transportation. But what about our buildings? How could we change really the spaces we live in? Well, we did some analysis a few years ago here in the MIT campus. And we looked at where we put energy in our buildings to heat them or to cool them and where people are.

We found, as you see in this graph, that there is no correlation whatsoever. We heat our buildings even when they're empty. Think about our homes. They're empty during the day, but we still heat them. Or think about our offices. They're empty during about the night. Nobody is there. But we still put a lot of energy in them.

So our idea was, what if you could actually use this kind of sensing and actuating in order to put energy only when we need it? We called the project Local Warming. Really think about not heating the whole space or the whole building, but actually creating a bubble of heat just around people.

We did the first test and experiment outside MIT at the entrance of 77 Massachusetts Avenue. You could actually step on the carpet, and then this kind of bubble heat would follow you, accompany you, as you entered the building. People loved it. Some of them freaked out. But then we thought that was interesting in the way that you can actually keep a baseline, which is colder, and have your own personal bubble that follows you inside a space.

What you see in the next video is actually an exhibition, an installation we did at the Venice Biennale just last year where we created a ceiling that could be installed in a building, in an office building. At the moment, it's very visible. We wanted to reveal the mechanism, but the whole thing could be made really seamless. And what actually this bubble of heat will supplement a baseline heating system just by following people. And here is the video.

[VIDEO PLAYBACK]

[END PLAYBACK]

We saw here how we can do this by making these bubbles of heat. Of course, you can think about the same thing also with cooling. It's a project we've been working on in Dubai. In Dubai, the problem is the opposite. It's that public space is very, very hot. It's very efficient to cool things with evaporative cooling. But then again, you don't want to do this across a whole big public space. So we looked at this as a way to create this kind of responsive mist that will cool the air just around where people are.

Another example of how we can use data sensing and actuating also in other contexts. We just did a project at the World Expo, finished a couple of weeks ago. And the World Expo was this year in Milan. It was before, five years ago, in Shanghai. In 2020, it will be

in Dubai. We were in a pavilion, and we were asked to design a supermarket, they said, of the future. See how a supermarket could change looking at the future.

And again, we started focusing on sensing on all the data we have now about products. What we did not want to do was this. It's a beautiful picture. Its a picture by Andreas Gursky. But it's not what we wanted to do. We found this an alienating place for buying things. What we wanted to do was use IoT in order to go back to the beautiful interactions we had in a traditional market.

Look at this. Products are on tables. Products facilitate interaction between people, and people between people and the things we buy.

One of the things we like was this quote by Italo Calvino, the Italian writer. And one of the Italo Calvino's characters one day goes to a cheese shop in Paris. And Calvino says that the guy thinks he's at the Louvre in Paris because every piece of cheese tells the story of a different civilization, of a different pasture under a different sun.

So the idea that today we know everything about things we buy. We saw it before as well. Big data applies to products. But then that information is not available usually when we are going to shop, when we are at the supermarket. Some people have tried.

In this example, done in Seoul, people tried to have a system by scanning products using a smartphone and getting some information about the different items. But again, if you look at this, it's a bit alienating. It doesn't promote interaction between people.

So our idea was, what if you could put again products on tables? A table is a beautiful interface that promotes communication between people. It's a collaborative interface. And then on top of the table, we got small scanners that scan how people move and what people are interested in. If you point to a product, if you touch it, if you approach it, and then magically, information then will come for that.

It's almost like an augmented label. Instead of having the little label on the product, you got much more real estate at the top to tell the story of what you are interested in. We like this idea of you approach an apple, and then this apple will tell you everything-- how many miles it traveled, where it was produced, farm, about nutritional values, and so on and so on.

And the more you engage with it, the more information you get in a dynamic way. If you only have a few seconds, you get the label, the bare minimum information. But if you want to stay longer, you can explore that more and more.

So there is actually the supermarket how it was built by our design office. You see here, once you make all the products visible, so it's slightly sloping down. So that from one point, you have, again, just visually all this information that you can capture. And then as you touch and approach, something starts telling you about all this information you might be interested in.

We also had to do some automated reshelving with robots in order to increase the density of products. But here, you can see the overall structure. And here you can see the finished project. You can see it is much more open space where people can play with products. A bit of robotics just for reshelving that helps to increase the density. You don't have the same density as in a traditional supermarket, where you got corridors. So you actually want to use technology to increase the density of items per square foot. And again, the space is completed in a space with people in it that, as they engage with the product, get to discover the stories of what they might buy.

And for us, it was a very exciting opportunity because the expo is really the ability to do something that will be tried by millions of people and to get a lot of feedback. In this case, it was really positive feedback about how this could enrich the interaction we have with the things we buy. Could lead us to a more informed and hence sustainable consumption patterns.

And the expo is also where we did another project. Just at the previous expo in 2008, in the city of Zaragoza, where we looked, in that case, at a different topic. In Milan, the theme was food. In Zaragoza, it was water. And the mayor came to us with this question. Water has been a beautiful ingredient of architecture and planning for thousands of years. How could we use it today in cities in a different way?

And our idea was imagine you have a pipe, and that pipe has many tabs, opening and closing controlled by a computer. Then you can create a living water wall that responds to people, a place where you can show images, or text, or patterns.

We got a commission to design the building at the entrance of the expo. We called it Digital Water Pavilion. The whole building is made of water. No doors or windows, but when you approach it, it opens up to let you in.

Inside, also, all of the walls expand and shrink based on how many people you have. The roof is also covered with a thin layer of water. And if you got too much wind, you can lower the roof to minimize splashing. Or at the end of the day, you can close the building, and the whole architecture disappears, hopefully without anybody underneath. We've got sensors for that as well.

When you look at this, it's really about how architecture itself, thanks to IoT, can become much more dynamic, almost like living architecture. If you look at this picture, it's something I like. You see the guy to the left had a trolley. He was going to the station. But actually, he stopped there to try to understand, what the hell is happening here?

What you see here is projections on the water. So it's about the physical pixels made of water and projections on the top, and a combination of both. What you see here is myself trying not get wet in testing all the sensors that detect people when they approach the building.

Now, I should tell you now what happened one night when all of the sensors stopped working. At that night, we were terrified. That night, the building will keep on doing its own crazy things, and cuts and holes and text and images, but without responding to people anymore. But actually, that night was one of the most fun nights ever. That night, thousands of kids from all over the city went to the building to play a new game. Not anymore a building that opens up to let you in, but a building that you need to engage like this.

[VIDEO PLAYBACK]

[END PLAYBACK]

And for us, it was an important lesson. Because as architects, as engineers, we always think that we know how people will use the stuff we design. But then reality, and especially human reality, and especially when the environment is filled with sensors, with IoT, with different feedback loops. Well, all of that, then, is a surprise. In this case, it was a good surprise.

So what we looked at today is really how IoT can help us to collect a lot of data in cities, but also to use these data to transform the space we live in. It could be about traffic. It could be about building. It could be about heating and cooling.

So all of these are examples of how this increasing, amazing availability of sensing and actuating in our environment is transforming the city itself. And the most exciting thing is really how all of this is put in the hands of people. It's put in our hands. It can actually empower us to make different and hopefully better decisions.

Week - 5 Module Five: Conclusion

Roadmap of IoT (Sanjay Sarma): Video 1

Welcome back. I am back now with a few slides to help you conclude this course, this journey that you've taken over the last few hours. This has been a fairly extensive deep dive into this vast area of technologies that refer to as the Internet of Things. And if you look at the slide that I showed you right at the beginning, I said there are a number of component technologies. There's location, there's sensors, interfaces, devices and so on.

But there's also system design issues. There's networking. How do you network these things, wirelessly, back hole, et cetera? There's architectural issues. There's security. There's data management. And we showed you what I think is a fairly dazzling range of applications, whether it's Smart Cities, or Smart Buildings, Smart Homes, and so on. And frankly, we've only scratched the surface, because when you start thinking about this, you will find yourself looking at almost everything differently.

In fact, I'm going to give you an exercise. Here's the exercise. Ask yourself the question, what would happen if my chair were connected to the internet? What sort of apps could I write with it? And it might seem like a trite, silly question. Why on earth would my chair be connected to the internet? But actually, think about it.

Perhaps it can tell you how much you're sitting down. Perhaps it can weigh you and tell you how much weight you've lost. Perhaps it can look at your posture. So right there, with something so trite, almost laughable, such as a chair connected to the internet, by asking yourself a question we have teased out, hopefully, some thoughts.

And I'm hoping that you'll get into this way of thinking. It's almost a way of thinking more than it is a technology. And that's what I'm going to now conclude with, which is, where do we go from here? One of the things we talked about is that there are many communication strategies when you do the Internet of Things. You have a stack of things. You have a device, you have some sort of Edge Computing, and you may have Cloud Representation.

And the way that you connect could be device to device, D-to-D, or what some people call M-to-M. As I've said in the beginning, and hopefully what you've gotten out of this course is, yeah, it's possible to do that. But really, why would you? It's a little bit overrated. Frankly, if you want to connect two devices, you probably want to bounce it off a gateway, Edge Computing, because something has got to make sense of these devices and the information they give to you, and then act on them.

And device-to-device communication sort of seems to make sense because that's how human beings communicate. But it may not be the right thing in the Internet of Things.

And then, you could have a device, to gateway, to gateway, to device. And that will happen, for example, if two devices are out of range from each other, or if you're bridging two different networks. So there will be a lot of stuff in bridging. Bridges is going to be a part of our future.

And then finally, what I think will be a dominant paradigm is when you go from a device, to the cloud, down to the device. And I call that DCD. And the idea there is, either you go to the cloud and then back to the device, or you go to the cloud, talk to another cloud, and it talks to its device. So for example, I get into a car, like Tesla for example, it's connected to the cloud. And when I drive to my home, my Tesla cloud talks to my home cloud and triggers certain actions.

For example, turning on the heating. I even joke sometimes that my Apple Watch or my Fitbit creates a cloud me. So you can imagine my cloud me climbing into my cloud car and driving to my cloud home, as the real me gets into the real car and drives to my real home.

So we've talked about all this. This is sort of just a way of thinking about some of the range of possibilities. By the way, if you insisted that that communication happen with HTTP, then that is called a Web of Things. It's a term of art. It's a sort of a philosophical approach to connecting things up. And there's actually a W3C group looking at it. And the idea is, look, we have the world wide web, we have HTTP. Why not just use that for communication and sort of inventing new stuff. So just want to put it out there.

So with that, let's talk about the reality of where we are today in the world of the internet of Things. Today, unfortunately, we are in a world of walled gardens. What does that mean? NEST, Home Kit, Smart Hub-- these are independent ecosystems. And this is the same in the industrial internet. NEST and Home Kit and Smart Hub are sort of home oriented. But really they're beautiful gardens, so they work beautifully, but there's a wall around them.

And we are inventing ways to sort of patch our ways across these walls. That's fine because we're in the early days, and a walled garden means that you can have an ecosystem that works. And things have worked out so far, and slowly the gardens are expanding. For example, NEST now can work with devices that are not made by NEST. And similarly, Home Kit is trying to sort of embrace other devices.

But frankly, it's expanding at the expense of the device manufacturers. And what I mean by that is NEST says to JawBone, hey listen, we want to expand. Why don't you work with us? So JawBone works with Google NEST. And then, Home Kit says, hey JawBone, why don't you work with us? And suddenly JawBone is like, I got to work with both these groups, so they end up working with them differently.

What we don't have, really still, is real interoperability. It's the vendors trying to sort of force themselves to work with two ecosystems, two walled gardens. But when the gardens start talking to each other, that is when this whole field will catch fire. And we

have to figure out when that is going to happen. That's really the tipping point, and that's what we've got to keep our eyes open for.

What I do is, I read all the blogs-- TechCrunch. I'm part of a lot of the standards activities. And I sort of recommend you do the same and really keep an eye on it as this field evolves. One of the things, one of the parts, I want to leave you with as we wrap up this course is, this is an evolving field. There is no answer. Don't let anyone convince you for the next five years that there is that answer. You-- and I'll talk about this more-- need to figure out the answer and draw it out to fit your needs.

Roadmap of IoT (Sanjay Sarma): Video 2

So let's think about this whole system-- what's a roadmap, what is some time, some predictions? Well, again, if you look at it as a stack, you have the device platform, you have local communications, you have some sort of gateway and edge computing, and then you have cloud. For example, in the device platform space, recently ARM released something called the mbed platform, which is promising.

But really it's going to take a couple of years for that platform to play out. And whether it's a standard that everyone agrees to or there's a dominant player, that's still got to be worked out. But there are some good steps in that direction. My view is it'll evolve. It's already beginning to happen, but it will keep changing all through the next couple of years.

Local communications, I think that will begin to mature in 2016, 2017 time frame even. And we might end up with a tribal situation, a BLE, a Bluetooth Low Energy tribe and a ZigBee tribe, or something like that. But that's going to happen a little bit quicker just because there's a lot of chaos there. And there are some new efforts beginning to form, for example, the Thread initiative, which has a lot of backers.

I think that the gateway slash edge computing issue that remains hard to standardize. And until that happens, you've got to pick your flavor-- which one do you work with? And that will take 2017, 2018.

Now, if you just take utility cloud computing, I would say that we're nearly there. But if you say cloud computing that works with smart edge computing, it's still walled gardens. And for that to standardize again, I think we're a couple of years away. So this is sort of where what I see is the evolution of the key components of the Internet of Things.

Now, I'm just going to share some personal biases. As I've said to you before, I believe that web standards are the way to go. Why invent the whole world?

If you use HTTP, Web of Things, for example, if you use Mashups, things are easier. We know how to live in this world. It's an easy design metaphor. I'm a big believer on that.

With some disclosure, the author of this book, Dom, is a former student of mine. I should say that there's a great book out, very inexpensive, called Building the Web of Things. It's a great way to get started in this space. So if you're interested in just building something, which I'll talk about next, that's a great way to get started.

And you know my recommendation is don't just do HTTP, lift things to the cloud. What I mean by that is take the real world and create an avatar world and then make the connections in the cloud. It's just easier, its cleaner, you get a level of indirection, it's more secure, and so on.

And then, finally, over time, I think that what's going to happen is we're going to go to a three-tier architecture. You have the device, you have the cloud, and you have edge computing, if you need performance. And that's really my prediction for where this world is going to go.

So you might be asking yourself, I work in a company, what should I do? And in that I'm going to quote the amazing philosophy of [INAUDIBLE] Nike, which is "Just Do It." Do something, get going because inaction is not a solution. You see, whether you like it or not, if you think of that example I just gave you, where I said, if my chair is connected to the internet, what can I do?

IoT is in the future. Devices you buy will be IoT-enabled. Your homes will be IoT enabled. And it's going to become a competitive thing.

And so what you need is what I call IoT literacy. It's a way of thinking, which is how do I instrument and take advantage of it because is happening. Don't fight it, in fact, try and win it.

Just imagine if you had fought the cell phone 10 years ago. If you didn't use your iPhone, your Android phone, or your Microsoft phone. If you didn't do text messaging. If you didn't do scheduling on your phone. If you didn't use Google Maps or Apple Maps, just imagine, you would have been at a disadvantage.

I would use the same thing. I mean if you have a factory that refuses to monitor valves using connectivity, compared to a company that has a factory that does. And if their insurance goes down, you're at a disadvantage. So it is in your future. I predict it. And don't fight it.

But it is very personal to you. What I mean by that is when we bring the technology in, let's say a cell phone. The cell phone is very personal to me. I use it in a way that is different from even a close colleague of mine. For example, I may use certain features more than she does.

My wife and I use our cell phones subtly differently, but within our family we have a certain pattern. We know how to reach each other. We prefer text to a call. IoT is like that.

If your business is your family, you will adapt IoT to your business. Your business probably has an advantage-- you do something different and it, gives you an advantage. So IoT has to wrap itself around that, so that you can use IoT to make the thing that makes you different more advantageous.

And so you have to figure out how to use it. Now, I'm not saying don't work with consultants. But if you work with a consultant, work with a consultant who understands the process. The IT part of it will come later.

But if you start with the IT, you will put the cart before the horse. The IT will dictate what you should be doing as opposed to the process. So figure out your process and figure out precisely where IoT can help you, then let's figure out the IT. I'll come to IT in a minute.

The next thing I recommend is build a real system and try and use it. I assure you the learnings will be fundamental. And it will give you a very gut-level, visceral IoT literacy that you will need.

And here's the next thing-- be ready to fail, be ready to iterate just as you would with math, just as you would with a new technique, just as you would if you, for example, decided to go buy a bike, and you've never ridden a bike. But you'll figure it out. It's the same thing. You got to learn to iterate because, again, this is going to be deep in your use, and you've got to figure out how this thing works.

Roadmap of IoT (Sanjay Sarma): Video 3

So if you are a beginner, where do you start? And my suggestion is, start small and be patient. So let's talk about that. Pick a single but complete example pilot. What could that be? Put yourself in different scenarios.

So for example, let's say you have a factory. And when you have a factory, perhaps there is a tank with water or some other liquid that you have a hard time telling whether the tank is full or empty. When it empties out, there's a problem. Wouldn't that be a great, clean example with tangible value? Find that example.

If there are machines that aren't getting turned off at night, is that something that you can fix easily? It probably is. You can have a very inexpensive IoT power meter that tells you that those machines were not turned off.

But make sure you pick a failsafe application, though. Because make sure that because this is your learning curve, that whatever happens sort of as backup, should, for example, the tank detector not work, you have other mechanisms to deal with that situation.

Also make sure you instrument for measurement, but in addition, for controls. Because you need to get into the mentality that when you measure something, you can actually act on it. Now the acting can be a display. It can be an alarm. So for example, when the tank is empty, a small chime sounds. It could be a red light that blinks.

But it's interesting, one needs to get into that mentality of taking information and acting on it. And the reason is it creates an appreciation for the sort of edge intelligence. You need it creates an appreciation of the latency, if you should go to the cloud. And you need to develop those instincts.

But the key thing is that it is important to start thinking about this from the very beginning. Now if it's your factory, you and your plant manager understand the factory. If you're a product company, for example, let's say you're considering introducing an IoT product, then you need to understand the customer, and the customer is an important partner. But whatever it is, it is your world.

Now I say this because one of the questions that comes up is, so where do the insights come from? The insights are actually an amalgam of your market insight, your insights about your customers, your insights about your factory, about your shop foreman's insights about the performance of the line with the technology.

And so you really have to be patient and search for those insights. But when they occur, that's when the value starts getting created.

So how do you implement something like this? One of the things I recommend is try and do it internally. Now I'm not saying that you will always have folks internally. And sometimes actually, your internal folks have been thinking differently. They've been

thinking about enterprise architecture. So it's not necessarily the enterprise architect who needs to do this. Maybe you want to hire in a new engineer. Maybe you want to hire in someone from the outside.

But try and keep it inside in the beginning, not to be secretive-- although that may be an issue-- but because the learnings need to be internalized. They will become part of your life blood. It's like a family thing. You need to understand how you're going to use. It's your ultimate personal implementation of the technology to make what you do differently more valuable.

Pick a non-realtime application initially. Because initially, you don't want to spend a lot of time struggling with realtime connectivity, scheduling, et cetera. So you want to take something a little bit non-realtime. That's why I used the example of the tank. Because a tank doesn't deplete in a microsecond. The threshold is reached, give yourself a little bit of a gap, turn the light on. If it's delayed by a couple of minutes, not a big deal.

My recommendation is use off the shelf technology when you're stumped. For example, a Raspberry Pi or an Arduino. A lot of these hobby systems are surprisingly powerful. But more importantly, they let you pull it out very quickly. And one of the principles of agile and lean is you want to go through as many prototype cycles and learning cycles as possible. You don't want to build that ultimate system. Because frankly, you won't. You will go through the cycles. You might as well learn.

One of the things is, well, should I use ZigBee or should I use Bluetooth? Use wires. Why deal with that issue in your prototyping phase? Deal with that afterwards. Use wires. You know it's a prototype. You're in for the learning. And that will get solved eventually.

And finally, I would recommend trying to implement for the cloud. And there are many reasons for it. One reason is that you will end up reconfiguring stuff. It's usually easier to reconfigure stuff if you're configured in the cloud. You can also log data for analysis. You may have some insights from that.

And the other thing is-- and I'm referring back to the book I mentioned earlier-- you can do mashups, there's very simple ways to implement a system if you use the cloud.

What if you're a maturing user? In other words, this isn't the first time you've used the Internet of Things. Maybe you've done stuff already. Well then, there's a different set of questions you want to ask yourself.

The first question, to me, is have you assessed your threats? Have you considered security? And I have to tell you that I am constantly amazed at the holes that I find in implementations. And to hope that something bad won't happen is not a great place to be. Because unfortunately, when it happens, it can be pretty bad. So please, I recommend thinking about security.

The next thing is when people, when companies and individuals implement IoT, they go through the honeymoon period, which is, you've implemented, everything's great. And for the first six months, nine months, one year, everything's great. Eventually, something's going to break. Eventually, for example, the sensor on a valve will stop working. Eventually, the valve might break.

Now the question is, do you have a maintainable system? When you're at the end of the honeymoon period, do you have a system that can be efficiently and sustainably maintained? So for example, if you send a plumber or a technician to fix the valve, let's say you have to change a sensor on the valve. Well, when you change the sensor, the sensor needs to be initialized as a key management problem. So now, is your plumber also an IT specialist and a security specialist? This is where your architecture becomes important. So think about that next wave.

The other thing you want to think about is , let's say that that sensor gives you faulty readings. Are you going to make faulty decisions? Could that run away from you? So you have to start thinking about these questions. How do you make the system fail safe?

Another thing is, can you update your system? Often, they build systems assuming that you've got to be there to fix it. With cell phones, for example, now you have over-the-air updates. And you have to think through, can your system be updated remotely? Because again, there'll be reasons for doing it. It may not just be technological. It can be regulatory. Maybe regulation changes and you've got to update it. And so you have to think that through.

And that's what I mean by the end of the honeymoon period. Honeymoon's ended, the couple comes home, and now someone's got to figure out who is going to wash the dishes? And this is what I would think about. These are just examples, if you're a maturing company.

And finally, what is your architecture? Can it be expanded? Can you add more features? Can you change features? What if a customer says, can you do this, can you do that? Have you locked yourself in? Have you painted yourself into a corner with your architecture?

Roadmap of IoT (Sanjay Sarma): Video 4

Another avenue of looking at this is what if you are product company? What if your company, for example, makes internet of things enabled faucets or internet of things enabled switches? Then the question becomes, where in the ecosystem do you want to play? Are you a device vendor? Are you a comms vendor, communications? Are you an edge intelligence vendor? Are you a cloud vendor? Do you want to be walled garden or do you want to embrace all?

Recently with Phillips, have a little hub for controlling their hue bulbs and there was some confusion because that used to be open. You could control other things and there was a sense that maybe they wouldn't, that Phillips would not let other devices be controlled. There was a backlash. So when you get into this space, one's got to figure out how you play in the space. And so where in the ecosystem, what are you? Sort of an existential question.

What architecture should you use? And similarly to my previous slide, security, can it be updated, is it expendable? This is very important. So finally, you got to think about expandability and what I mean by that is, you put one product out in the market and now you want to put another product, and you want these two products to work together. But what if you painted yourself into a corner and you didn't even have hooks in the product that you introduced to enable this other product to work?

A great example of expandability is Nest, you know, has the thermostat and now they have the alarms and the ability to get them to work seamlessly. Two Nest systems can work very well together. This is sort of the thinking that one needs to do if you're a product company.

And then another sort of fundamental question I think that one needs to think through is, what is your business model? You might think in an internet of things class, which is a technology class, business models ought not to be considered. But I actually think they're very fundamental because in this new world, do you sell them for purchase? For sale? Do you have subscriptions? Is your device simply leased out?

By the way, it's not that new. If you have a Verizon set top box, then that's actually leased out. You don't buy it. Do you run it off advertising? There's many, many different models and those are just three. Here's another one. Do you deploy the model, the technology to, for example, draw your revenues from insurance? So for example, a home alarm system, it reduces insurance.

So all these business model questions become possible because of the connectivity of the technology, but also a connectivity and data sharing across businesses, assuming that you have the go ahead and the permission of the customer. So there are some pretty existential questions and to assume that your business model with which you worked and operated all these years will translate into the internet of things world I think is, you're missing a trick. You've got to sort of think that through.

So with that we come to the end of this course. This is an evolving technology, as I've said before, and this is a space that I think we will all live in within the next few years, and many of us already live in it today. I hope you found it useful. The concluding message that I leave you with is just do it. Do it thoughtfully, but just get going. Thank you very much.