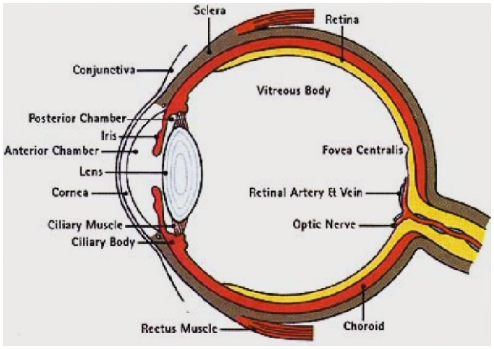
3.1 Vision as a Computational Problem

[00:00] Today we're going to be looking at yet another aspect of the ways in which human minds or animal minds and what we might think of as computational minds - the two perspectives can lend insight into each other. And this is a particularly interesting aspect of that discussion as we talk about vision. So what we're going to be looking at today and over the next couple of lectures will be the ways in which we can treat human vision - again, with occasional forays perhaps into animal vision, but primarily human vision - the ways in which we can treat that as a computational problem.

[00:56] This perspective, that is viewing vision itself as a computational problem, is a relatively new idea. You might say that it actually started with interest in machine vision in the 1970s or thereabouts, and it has grown to an extraordinarily fruitful enterprise. That is, the study of getting machines to see objects and interpret scenes, sometimes generally, the way that we imagine people do. Sometimes the intent is to get machines to go further than people are able to do. But for now, we'll start with the ideas of what we know about human vision, and how those can be treated in a computational way. It's a really fascinating story. For our purposes, we'll begin with a very rough picture of what human vision does. The reason that we're going to do that is because it simplifies the discussion of the computational treatment of vision. That is, if we start with a rather simple view of what human vision does, we can get some foothold on how to treat it as a computational problem. But we'll begin then with this rather simplified portrait of vision, and really this diagram which I got off the web.

[02:34] This is a diagram of the eye, and it includes much more detail than we really care to use in our discussion. For our purposes, we can think of the human eye as being something like a pinhole camera. Now, in a sense, that's already a kind of machine metaphor for the eye, if you want to think of it that way. The eye has this red small hole through which light comes, is refracted by the lens in the eye, and then the focused picture is projected on the surface of the retina. So you see that in this diagram here the retina is that, my hand disappears behind the picture, but you see the yellow layer there: that's the retina of the eye. So to a first approximation, think of it this way. The light comes in from the scene outside our heads. It is projected onto this essentially two-dimensional screen at the back of the eye, the retina. Now, for our purposes again, there's much more to say about this, and there's a lot more detail. But the first thing to notice about this very rough portrait is that what we have to do as people is then interpret the patterns of light falling on the retina to translate them into beliefs about three-dimensional objects out in the world. That's an extraordinarily difficult task when you think about it. Again, we feel that it's so automatic. We just open our eyes and see objects out in the world. But when you view this as a true computational problem, you see that it's anything but automatic. In fact, it's fiendishly difficult. What we're trying to do is take patterns of light falling on



an essentially two-dimensional surface at the back of our eye, and from those patterns of light we're going to generate beliefs, well-founded beliefs we hope, about where objects are in the world.

[05:00] One thing to say about this, right off the bat, is that from a mathematical standpoint, recovering three-dimensions from a two-dimensional projection is impossible - in the most general statement of this. That is to say, if you are looking at a two-dimensional projection, like a scene on a movie screen, if you are looking at a two-dimensional projection of 3D objects, the three-dimensional objects that could give rise to that two-dimensional projection are infinite in number. You have to do a great deal of interpretation to imagine, to choose a particular set of 3D structures that is generating the particular two-dimensional projection that you're dealing with.

[05:58] This is an example - this visual illusion by Shigeo Fukuda, Japanese artist, gives you a suggestion of what I'm talking about. This is a photograph. What you're looking at is a sculpture with a whole bunch of weirdly placed pieces in front of a mirror. The mirror is reflecting what looks to be a grand piano. It isn't reflecting a grand piano: it's reflecting that jumble in the front part of the photo. In other words, if you were standing behind the mirror, if you are looking at this object from the position of the mirror, you would see a pattern of light that you would justifiably interpret as a grand piano, 999 times out of 1000 or even more than that. If you're seeing something that looks like a grand piano, it is. But it doesn't have to be, it might in fact be that jumble of pieces. This is what I mean by saying that the general computational vision problem is mathematically impossible. That is to say, let's go back to the earlier slide. What you're trying to do is take this pattern of light on the two-dimensional retina and infer from that where the 3D objects are. In point of fact, an infinite number of possible arrangements of 3D objects or pieces could have given rise to that very same image.

[07:44] What does that mean computationally? It means that this is what we're going to have to do in order to solve the vision problem for ourselves when we open our eyes. What we're going to have to do is make a lot of guesses. Those are educated guesses. They've been honed into us through millions of years of evolution. So we're very good at making these kinds of guesses, and generally, we don't encounter scenes like this. So we don't have to worry about being fooled by a collection of 3D objects out in the world that are arranged to look like something familiar. However, just as a general rule, it's good to think about illusions like this. Illusions - optical illusions, visual illusions - they're often very good hints as to the computational nature of our own vision. That is to say, they tell us something when we go wrong. When we look at something that is an illusion, or we're confused by it, that often gives us a clue as to what algorithms we're running to interpret scenes.

[09:10] Here is an example. This is a photo by the wonderful photographer Walter Wick showing an impossible dog house. So you might look at this photo and try to imagine how something like this could be. Again, this is a photograph. So this is a real object in the world.

You look at this thing, you have to decide for yourself, what's wrong here? I'm having trouble interpreting this as a true three-dimensional object. The trouble that I'm having should be a clue again to the algorithms that I'm using to interpret two-dimensional scenes as three-dimensional objects. In this case, you might look at some of those boards in the dog house, and when we see a portrait like that, we interpret things, for example, that look like they're solid or connected as in fact solid or connected. It's rare that things that appear to us as one piece aren't, but it can happen. This is another view of the dog house from a slightly different angle, and you can see how this is a much clearer version to us of what the three-dimensional dog house looks like. But when seen from a particular angle, it looks like things are crossing in ways that would be impossible in three-dimensions. So the list of wonderful optical illusions - there are books and books of them - they're so amazing and often beautiful to look at, and they teach us something. Many of them teach us something about our own, I will say, algorithms and vision. That is they show us the limitations of our own strategies for interpreting two-dimensional scenes.

[11:22] Now, what we're going to do in our ensuing discussions of vision, we're going to kind of begin with a very simplified version of the vision problem. So for those of you who are watching this who are computer programmers, I'll put this to you as a kind of computer program, an assignment of a computer program to write. This would be a very difficult program to write clearly. The idea is you are given a photograph, say a grayscale photograph like the one that you see here on the left, okay? You're given a grayscale photograph. Now, that's translated into a large number, an array of pixel values. A pixel should be a familiar term, but it's a little element of a two dimensional graphical picture. In other words, think of this photograph at the left as being say 1,000 by 1,000 pixels, 1,000 rows, 1,000 columns and therefore one million pixels. And each pixel is, in this case, a number which can range from, we'll make it an 8-bit number, which can range from 0-255. And 0 corresponds to black, and 255 corresponds to bright white, and everything in between corresponds to some shade of gray. So what you're looking at in this diagram is a photo which would be perhaps a million pixels: 1,000 by 1,000. But if you see that little - I think I can just point to it - this little box here that's at the side of this little black region, this rectangle that's been drawn at the side of this cube, that's expanded into the set of numerical values that you see at the right. So the particular numerical values are not so important here. But if you look carefully at them, you'll see that they range in value in that range from 0-255. So here's our computer program problem. We're given a 1,000 by 1,000 array of numbers just like the little subset that you see at the right there. So you're given a 1,000 by 1,000 array of numbers, all ranging from 0-255. That's your input. Your output should be something like “hand holding a cube”. In other words, the program that you're going to write is going to be one that takes as its input pixel values, which as our first approximation are the intensity values received on the retina. So your job is going to be taking that very large number of intensity values. In this case, we're not even dealing with color, we're just dealing with grayscale. So you're going to get these numbers, and you have to interpret that as a particular

three-dimensional scene; a hand holding a cube.

[15:03] That is a very difficult computational problem, and you have to make lots and lots of assumptions about it. I should say off the bat that there are lots and lots of simplifications about this statement of the problem relative to human vision. We do have many more cues than simply an array of numbers like this. For one thing, we have two eyes. So a better statement of the problem would be that we have two corresponding scenes of two corresponding arrays coming from slightly different vantage points, and that would help us interpret the three-dimensional structure of these numbers. Our eyes are also, it should be mentioned, the retinas don't return arrays of pixels which correspond to an evenly distributed interpretation of the scene. In fact, the resolution of our retina is far greater in a central area called the fovea. So if you were representing the human eye in this array, to be more accurate, you would show a very high resolution chunk of numbers toward the center and a lower resolution chunk of numbers towards the periphery. That's another approximation that we're making. Here we're just saying, okay, we're just going to assume that we've got a 1,000 by 1,000 array so the same resolution is true throughout the scene.

[16:48] Of course we can do other things. We can move around. We might integrate our understanding of the scene by reaching out to touch certain objects. We have color. Things are moving. We can move. The objects might move. There's all kinds of other cues we might pick up. Still, this is not the most unfair portrait of the vision problem because when you think about it, you were able to interpret that two-dimensional photo on the left. You didn't have color. It didn't matter if you moved about. The photograph is not moving. In other words, you're able to look at that array of pixels on the left and generate the response that this is a picture of a hand holding a cube. We would like to be able, again as a first approximation to the vision problem, we'd like to get a computer, a program, to perform in the way that we do as people. We'd like to get a computer to look at this array and make a well-justified guess. It's all it could be as to what the three-dimensional objects are that are being represented. So that will be our starting version of the computational vision problem and that's what we're going to use as the basis for our discussion. We'll expand things a little bit as we go along, but that's what we'll use as the basis of our discussion as we continue talking about vision.

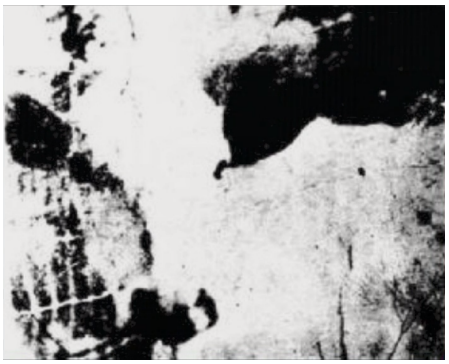
3.2 Finding Edges

[00:00] We've begun talking about treating vision as a computational problem. Our rather rough-hewn first approximation to the vision problem was expressed as this idea. We're going to imagine that we can write a program that will take arrays of pixels, which are just arrays of numerical values corresponding to intensity. In this case, we're not even dealing with color, we're just dealing with gray-scale images. But what we want to do is write a program which will take a large array of pixel values. In our first version, we stipulated that the pixel values will be 8-bit numbers between 0 and 255, where 0 corresponds to black, 255 to bright white, and everything in between corresponds to some shade of gray. What we're going to do is take that array and interpret it as a three-dimensional scene. As we noted already in our discussion from last time, this is a mathematically impossible problem. That is to say, an infinite number of potential three-dimensional scenes could give rise to the very same array of pixels. Therefore, what we have to do is make guesses about the nature of 3D objects in the world, what kinds of things we might be likely to see, and we have to employ those guesses in writing our program to interpret these arrays. There's no alternative to it. It could be that this particular array of pixels has been generated by some wild jumble of points out in the world with different intensity values. That's possible. It's extraordinarily unlikely. So one of the things that we begin by doing is making assumptions about the nature of objects in the world; what's likely to generate arrays of intensity values like this.

[02:28] A simple assumption that we all make, that animals make when we open our eyes and are trying to solve the vision problem for ourselves, is that the world largely consists of solid objects with boundaries. So our first job in looking at an array of pixels like this is going to be to try and find where the salient objects might be. What that translates to in programming terms, in looking at an array like this, what that translates to is looking for edges. We're looking for edges between white and black or gray-scale and black. In effect, what we're looking for are very sharp transitions in intensity. So I mentioned last time for instance that little, I'm trying to point to it with my finger, that little rectangle there - that rectangle includes in it an edge. That is a place where the pixel values rapidly go from the white of the cube to the black of the background. That would be a cue for us, if we're looking at this array of pixels, that would be a cue for us that rapid transition, that there might well be, we can't be sure, but there might well be a three-dimensional boundary there. There is a boundary of a three-dimensional object. So our first step in looking at a scene of this kind is to try and isolate where the interesting edges might be. Again, we take into account that there are all kinds of ambiguities and potential errors in doing this, but we're going to make a succession of guesses at what the object might be.Naturally, we have to begin someplace so let's begin by seeing where we think the edges of the object might be.

[04:44] Now, we do this. It's very clear that we have difficulty interpreting an image if we are trying to find where the edges are. This is a famous, I'm not sure I could call it an optical illusion, it's a famous example of an object that is difficult to interpret if you haven't seen it before. It's interesting. If you have seen this, this is a photograph, if you have seen this photograph before and interpreted it, then you won't have any trouble re-interpreting it the next

time you see it. That's already actually something quite interesting about our vision, about human vision. If you have not seen this photograph before, you may have interpreted it right off the bat or may not. I will just tell you that this is a picture of a cow. Now, again, if you have never seen this before, you may well be looking at this picture and thinking, what cow? That was my experience when I first came across this in a book, and the caption of the photo was just a cow. I don't know, five minutes or something, I was staring at this thing and going, what cow? There is a cow there. This is an advantage of the video format. What I would urge you to do, because I'm about to spoil the illusion for you, so what I would urge you to do is pause the video at this point and try to see the cow in this photo. It may take you a little bit of time if you've never seen it before. But look for the cow. Do that now, and when you restart the video, I will show you where the cow is in the photograph.



[06:54] I'm assuming that you've paused the video, and now we're looking at the photo. Hopefully, you've seen the cow. If you haven't, I'm about to spoil it for you. I'll see what I can do with my finger here. See, here's one of the ears of the cow and there's the other ear. Hopefully, you can see her nose, going around there. Then you can see the side of her head, and here is my finger right in the middle of her right eye. Now, hopefully, the cow is jumping out at you.

[07:33] Why was this picture so difficult to interpret? It's because the edges aren't very clear in it. So that very first step that we're incorporating into our program, the step of looking for boundaries of objects, is particularly difficult for this photograph. Now, interestingly, as I say, once you have interpreted this photo as a cow, you can't unsee it. You can't look at the photo and be confused by it anymore. That's a very interesting cognitive fact about vision. Even more interesting is the fact that if you see this photo perhaps a year or two from now, it won't be a problem for you. You will have visual memory of the photo, and you will remember interpreting it as a cow. So if I can call it an illusion, this illusion is now gone for you forever. But there are things to think about in those facts as well. For now, we'll sweep those under the rug and just deal with the issue of finding edges in a gray-scale photo like the one before or like this one. How would you go about finding edges?

[09:05] Now, we're going to use a technique that should be familiar - if you've taken a course in Signals and Systems or Signal Processing, then you may well have seen a technique called convolution. We're going to be using a convolution process to find the edges in images. I'm not assuming that you've seen this term before, and I'm going to do my best to explain it here in lecture, and there will be additional materials that I'll provide so you can exercise your knowledge of what convolution is. But to begin with, the purpose of convolution is to take a picture like, again, I've gotten this from other sources off the web, but the purpose of convolution is to take an image like the one at the left, which happens to be a full-color image, and to produce from that image a collection of lines that probably correspond to object boundaries or at least points of interests. In other words, you could view what we're trying to do at the beginning as vastly reduce the complexity of the problem. We still have all the information present in that

photograph at left, but what we're trying to do initially is just eliminate lots and lots of detail at this point and see if we can just find where the major objects are that we're going to have to interpret. That means finding the edges of those objects. So convolution is the mathematical procedure that we're going to be using to take the image at the left and produce the convolved version at the right.

[11:04] Now, a lot of detail and a lot of potential information is being lost in getting to that point. Again, we still have it on hand if we want to go back and look at the original photo, but what we want to do is take that image at the right, and see if we can just begin to find where the objects are. This is the first step in image processing, and we'll view it as one of the first important steps in vision. In fact, edge detection is seen as one of the major tasks of what's called low-level vision. Low-level vision, meaning that it's essentially unconscious, automatic. You can't stop it. Just like in that picture of the cow before, once you have seen the edges in the cow, you can't unsee them. So edge detection is not a process completely under conscious control. It's largely automatic and it tends to be very fast. In cases like the cow, when it isn't very fast, we notice it. But in most cases when we open our eyes, we're not actually making any great effort to see edges in the world. That happens swiftly and relatively automatically and unconsciously.

[12:39] So what's this magic process of convolution? Here's the idea. Think of that photograph on the left as being a scene laid out on the floor, a big two-dimensional image that's like a mat photo that's been laid out on the floor. What convolution does is, here's the basic idea. You're taking that original image and processing it in a way to look for these rapid changes in intensity. Now, specifically, what you're doing for our task of finding edges is convolving that image with what's called a sombrero function. Now, I'll get to the meaning of convolve, but let me just show you what this sombrero function is. I think of it more like an orange juicer, think of it as a big hump with a trough all around it. The hump goes above zero. This is a numerical function so the hump is a positive value. Then there's a trough around that hump which has negative values, less extreme negative values than the hump in the center, and then it flattens out from that point on. So this is called a sombrero function. I sort of think of it as an orange juicer that is positive, negative trough around, and then essentially zero as you get further from the center. Now, this function, what we're going to do, is take this function and imagine it being like an object that I can place. I'm going to place that sombrero function over each and every pixel of the image. So if you like, you can think of what I have as a million copies of this sombrero function that I'm placing over each and every pixel in the image.

[14:56] Now, what do I do with the sombrero function placed over a particular pixel? What we're going to do is, multiply the central pixel by the center of that hump and all the surrounding pixels by the values surrounding that hump. In other words, I'm doing my best with a sombrero function. Imagine this is a sombrero. It isn't, but it's my hat. So I'm placing this hat above, centered at a particular pixel in the image, and then I'm doing many, many multiplications. I'm multiplying the center pixel intensity by this value in the hat, and then the surrounding pixel intensities by the nearby values in the hat, and then the further surrounding pixel intensities by the other values in the hat. So a convolution is actually a multi-part step. You're taking one of these orange juicers, placing it above a pixel, doing many thousands of

multiplications, and adding the result of all the multiplications together to get one final value. So again, let me go through this carefully. What you're doing is taking this sombrero function, placing it above a region centered at a pixel but above a region of the image, doing thousands of multiplications and one big addition. The result of that addition is going to be the result of this convolution process at that one point, and you're going to do this at each and every point of the image.

[16:53] Now again, if you're a computer programmer, this may sound to you like, this is computationally very expensive, isn't it? For each and every one of these million pixels, we're doing a convolution which itself involves perhaps many, many thousands of multiplications and one addition at the end. That's an awful lot of computational work. In the extreme and most thorough version of convolution, that's true, but a couple of things should be noted. First and perhaps most importantly, is that each of these convolutions can be done in parallel. The result of a convolution at this point does not depend on the result of a convolution at a nearby point. So each and every one of these convolutions can be done in parallel. If you have, in fact, a million of these little sombrero functions, then you can do a million convolution steps altogether like a chorus. You don't have to wait to do them all in sequence. You can do them all at one time. Moreover, realistically, you don't have to do many thousands of multiplications. You can get a pretty good approximation to this convolution step by just doing a smaller number of multiplications, and we'll see over time how that works.

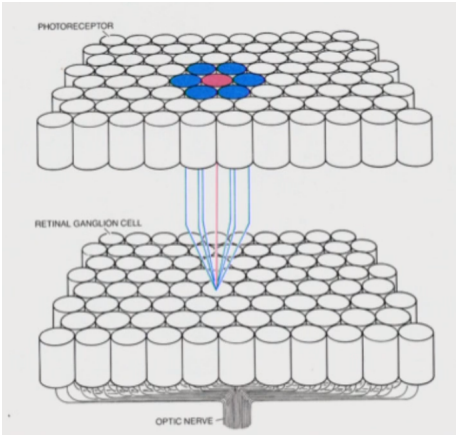
[18:27] Once you have done this convolution step, what you are then looking for are places where the result of the convolutions just go from negative to positive, or vice versa. The mathematical phraseology here is that you're looking for zero crossings of the convolution step. Once you've found the zero crossings, those are going to tell you where the edges are. So what you're looking at in that right image is the zero crossings of a convolution step.

[19:03] Now, this is again, it's a well-known technique from signal processing, and it's computationally actually fairly efficient. There are other kinds of advantages to it. For example, I could just mention a couple. You're looking for zero crossings. The slope of the zero crossing tells you something about the rapidity of the change in intensity, that's one queue that you have. Moreover, the idea of the sombrero function is that it's set up so that if you convolve an image which is uniform, that is to say, if you place your sombrero function over something that is pure white or pure black or pure gray, the result will be zero. The reason that that works out is because you can think of the hump value as having a strong positive bias, the trough as having a mild negative bias, but one that is all surrounding the central hump. Then the outer regions as making a very mild contribution to the result because they're close to zero. So what you're really doing is multiplying the center values by a large positive number, the surrounding values by mild negative numbers, and then the outer values by values that are so small, so close to zero that they're really not going to matter.

[20:55] There's not just one sombrero function. You can have an orange juicer that has a low but wide hump with a trough around it or a very sharp peak with a trough around it. So there is not just one sombrero function. You can think of sombrero functions as being tuned. A sombrero function that has a very high peak, it doesn't even look like a sombrero anymore. It looks like maybe almost a cone or something with a trough around it nearby. A very wide

sombrero function also doesn't look like a sombrero, it just looks like a mild hill with a little gentle valley around it. Those are extremes of the sombrero function and any one of those could be used in this convolution step. In practical terms, the mild sombrero function corresponds to a very rough view of the scene, where you're just picking up very large and important edges. A sharp sombrero function picks up more detail. It picks up edges that might be smaller or finer or corresponding to milder changes in intensity. So by using sombrero functions that are tuned to different kinds of edges, you can actually take the very same scene and, in a sense, look at it in a blurred perspective like the photo at the bottom left here, or in a sharper perspective like the photo, the image in the bottom right here. That is in both cases, we're convolving this original scene with a sombrero function. But at the bottom left, it's a sombrero function that is broad and blurry. It's like squinting your eyes almost and then at the bottom right, there's a sombrero function that is sharp and picking up more detail. So there's a lot of advantages to using this convolution technique. You can actually use it at different tunings to start by getting a very fuzzy version of the scene and then gradually incorporating more detail as you go along. This is a tried-and-true technique in machine vision: convolution with a sombrero function. What about human vision? Do we do anything like this? It seems like an awful lot of mathematics to be employing just to find edges in a scene. Do we do anything like this?

[23:50] The answer is, to some approximation, we do. In fact, our own retina, behind the retinal cells, there is a layer of what are called retinal ganglion cells, which in effect perform something like this convolution. Again, this is a diagram that I got from an article, but what you're seeing at the top are a hexagonal group of photoreceptors in the retina. The way you're supposed to interpret it is, think of that central red photoreceptor as responding to high-intensity, that is providing a strong signal. It's looking for high-intensity, and it will signal a high response if it gets a high-intensity right at that photoreceptor. The blue photoreceptors around it are looking for, they're sensitive to lower intensity, and they will contribute a mild negative response. That is a mild response if you have low intensity. So think of that entire hexagon with that red photoreceptor in the center. If that area is uniform, if it's all the same intensity, then this is acting kind of like a sombrero function. The red at the center will signal a particular value, but that'll be subtracted out by the values of the surrounding blue photoreceptors. In other words, the red photoreceptor is adding in a positive value for intensity, and the blue photoreceptors are adding in mild negative values for intensity. They want to see low-intensity. So the result at that ganglion cell in the second layer, if this little hexagon is uniform in intensity, the result will be zero at that ganglion cell. We'll get a positive value coming in from red, mild negative values coming in from blue, and the result will be zero when it gets to the ganglion cell.



[26:12] Suppose on the other hand, that this hexagon is positioned right over an edge that's going, let's say, right down. Imagine a line coming from the back of that array to the front and going right through the center red photoreceptor. So imagine that there's white of

high-intensity at the red photoreceptor and at the three blues to the left, and low intensity at the three blues to the right. That's kind of an approximation of what we're talking about: an edge that goes from white at left to black at right, and the edge is pretty much passing through or maybe is positioned just a little bit right of center of that red photoreceptor. Now, think about what happens then. The red photoreceptor signals, yeah, high-intensity. The three left blue photoreceptors are subtracting out mild values. The three blue photoreceptors at the right, however, are subtracting out very low values. They're looking for low intensity so they're not going to respond by subtracting out values, they're just going to essentially subtract out zero. The result then is that you have a positive value coming to that ganglion cell. The positive value is something that's going to be the contribution of the red cell minus just a few mild values from the blues at the left, and minus nothing from the blues at the right.

[28:12] Let's try a slightly different scenario. Suppose that the white to black edge is moved a little bit to the left, and the white edge is just passing over those three blue cells at the left, and then there's dark values at the red photoreceptor in the center and the three blue cells at the right. In that case, the red cell is not responding. There's low intensity right over the red cell, so it's providing a zero, essentially. The three blue cells at the right are also providing a zero, but the three blue cells at the left are throwing in a mild negative value, all three of them. The result will be that the ganglion cell gets a negative value. It gets a negative value because the three blue cells at the left are signaling high-intensity, which we don't like, and subtracting that out. The red cell is seeing low intensity and not supplying anything. The blue cells at the right are seeing low intensity and not supplying anything. So the ganglion cell in that case is recording a negative value.

[29:35] The purpose of this very intricate discussion is just to show that the mild positioning of an edge, a white to black edge over this set of photoreceptors in the retina can result in a shift from negative to positive in the ganglion cell in the layer right beneath. That's a very good approximation to what the convolution step is doing. It's taking a central value, which is positive, and subtracting out mild negative values corresponding to the intensity surrounding that central value. So this arrangement of photoreceptors in the human retina with a layer of ganglion cells behind them is similar in function to the convolution step that we've been talking about. This is an interesting case where thinking of vision as a computational problem can allow us to look for functions in the biological eye, and infer the purpose of those functions in the biological eye. So it makes sense that you would want very early in the vision process, as in this case just about as early as possible, you would want some techniques built into the eye that would respond to the positioning of edges where they can. We'll continue on with other correspondences between the biological and computational vision problem as we go along.

3.3 Depth Perception

[00:00] In today's talk, I'd like to focus on one particular problem involving a computational view of vision. So we've already talked about, a little bit, one task of vision which is to determine where the edges of objects are. What we're gonna do today is focus on yet another aspect of vision. If you like, you can think of it as a computational problem that our vision system has to figure out. Which is not just where the boundaries of objects are, but how far those objects are from us. When you think about it, that's actually a rather difficult task. It's an important task because, again, what we're seeing is a collection of three-dimensional elements projected on the 2D surface of the retina. As we've already said, it's mathematically impossible to recover information like distance, at least from a single picture like that, from a single eye. We are pretty successful at it even though it's mathematically impossible, we're pretty successful at it.

[01:41] Imagine for example, you're walking on a street and there are people, and cars, and stores, and dogs, and all kinds of things in your line of vision. One way that you have a sense of how far or how close these objects may be to you, you have a number of cues that you're able to use. One being that you already kind of know the sizes of these objects. So if in your field of vision the human being looks about as tall as the dog, then you have reason to believe that the human being that you're seeing is a good deal further away than the dog is. The dog is closer up, the human is further away, and on the surface of the retina, they look to be about the same height. So from that cue alone, you can determine that the human being is probably a good deal further from you than the dog. Other things involve movement. When one object moves in front of another, it occludes or blocks that object. So that's another cue that you can use to say that one object is closer to you than another.

[03:04] What we're going to do is focus on one way in which the human vision system and many animal vision systems are able to determine distances of objects. That's by the fact that we have two eyes and can use stereoscopic or binocular vision to collect clues about how far objects are from us. The fact that we have two eyes gives us two vantage points on particular objects and that is one of the things that enables us to determine distance. It's not the only thing that enables us to determine distance since we know that people who have vision in only one eye are able to see and get around pretty well. So they don't need binocular vision to make judgments about the distances of objects; they don't absolutely require it. But as so often happens when you're talking about the human vision system, we use every trick that's at our disposal. Since most of us have working binocular vision, that's one of the techniques that we use to determine how far an object is from us.

[04:20] So let's begin with a fairly abstract diagram of what we're talking about when we say a binocular vision system. There are a lot of little labels on this particular diagram so let me explain what's going on here. What you're looking at, those two ovals at this level, represent the left eye. You're looking from above the top of somebody's head, and there's their left eye and their right eye. Clearly we've abstracted away a lot of detail here. So both of the eyes are pointed forward, they're just taking in a scene in front of them.

[05:10] What we're now interested in is the distance from us. When I say us, I mean the plane between the left and right eye. So imagine there's a plane there that intersects, goes through the head and goes through the left and right eye. What we're interested in is the distance of that little blue point that you see at the top from the plane that goes between the left and right eye. So again, imagine what you're seeing in this diagram is a straight line drawn between the left and right eye, and imagine that coming out and into the plane of the diagram. That's the plane we're talking about. We want to know how far that blue dot is from that plane. That distance, which is the thing that we're trying to determine, is labeled D in this diagram, capital D on the right. Now, that line below the left and right eye, that horizontal line, think of that as the line of the retina, the screen of the retina. Again, just for simplification for this diagram, we're thinking of it just as a continual screen. So you have the left eye and the right eye and that line of the retina onto which the image is projected. The focal length of the lens of the eye is that distance F that you see at the right. So to sum up so far; we're interested in finding the distance D, we have two eyes, left and right eye, and they are at a distance F from the retina. The focal length of the eye lens is F, so they're projecting onto the retina at a distance F.

[07:15] Now, a few more labels here. If you were to draw a line directly from the blue point perpendicular to the retinal plane, you see that line drawn in the diagram, it would intersect the retina at a particular point, which in this diagram is between the two eyes. The distance from the center of the left eye to that intersection we've labeled L, script L, in this diagram. The distance from that point to the right eye we've labeled R, script R, in this diagram. The distance between the left and right eye, in other words, which here is the sum of L and R, the distance is L plus R equals this thing called B.

[08:12] Now, a couple of things to note here. F, the focal length of the lens of the eyes, is a constant. So F is not going to change. You could think of F as a known number that always stays the same as we walk around, same number. B, the distance between our two eyes, is likewise a constant; it doesn't change. Well sure, as we grow up and our head size changes, it changes. But to a very good first approximation, B doesn't change, it's just the same. It's the same from day to day for us. So B is a constant, F is a constant.

[08:59] Now, we're going to do a little bit of geometry looking at this diagram. Remember, what we're after here is we want to compute the distance D knowing that F and B are constants. So here's what we're going to do. Draw the line from the blue point to where it projects onto the retina. So that line is drawn as the diagonal in this diagram. Similarly, draw the line from the blue point to where it is focused by the right eye on the retina. So you see that we've drawn that as straight lines from the blue line to the retinal plane through the center of the left eye and through the center of the right eye. Now we're going to do some geometry that you may remember from high school. We're going to look at similar triangles, and here's the idea. So we're going to look at similar triangles here, and what I want you to focus on is the triangle that goes from the blue point. I'll try to draw this, so here's the blue point. Here is the left eye, and now we're looking at the retinal distance. Pick out in the diagram on the right here, these lines: it's the line from the blue point through the left eye to the retina here and the line that projects right onto the retina here. Remember that what we've got here is a distance called Alpha, and this distance is called L. Now we're going to look at similar triangles here. Notice that this triangle is similar to this

slightly larger triangle. Why? They're both right triangles. These two lines are parallel. If I draw a horizontal line from the center of the left eye to this line, I've got a line parallel to the retinal line here. So this triangle that I'm outlining with my hand is similar to this triangle that I'm outlining with my hand now. I want you to look at those two similar triangles on the graph here. So blue point to center of the left eye to the interior point of that vertical line, that's a smaller triangle, and blue point to the point on the retina where it projects down to the point between the two eyes, that's the larger right triangle. Those two are similar triangles. What that means is that this distance here is the one we're looking for, D. This distance here is D. The ratio of D to L, because these are similar triangles, the ratio of D to L is the same as the ratio of D plus F to L plus Alpha. So that's just high school geometry right there. The ratio of D to L is the same as the ratio of D plus F to L plus Alpha.

[13:47] So let's go to the next slide, and now I've written out the equation that we just found, the ratio D to L is the same as D plus F over L plus Alpha. You can do the same observation on the right side of the diagram, and you get D over R is equal to D plus F over R plus Beta. The rest is left to the reader. I will let you do the algebra to manipulate those first two equations, and what you can derive from them is an equation for D. The equation for D is that third equation written on the graph here. D is equal to F times B over the sum Alpha plus Beta.

[14:38] Now, what have we got so far? It may just look like we've done a lot of work, and we've gotten just some equation with a lot of numbers in it. But there's some very convenient things about this equation. The numerator F times B, remember both those numbers are constant, so the numerator on the right side of the third equation there, that's just a constant. I don't know what number, it depends what units you use and so forth. But once you've calculated it, it's a constant; it's not going to change. The bottom quantity Alpha plus Beta, that is, look at the distance Alpha on this graph, and then Beta on the right side of the graph, the sum of Alpha plus Beta is called the disparity. So what we found is that the distance of a point from the plane of the eyes is inversely proportional to the disparity. If you have a very large disparity, you have a short distance. If you have a small disparity, you have a large distance. So this is a formula, and as a first approximation, this is a computational approach that we can take to using two eyes to determine the distance of an arbitrary point. Given a point out in space, we look at it with our two eyes. We internally note, the computational elements of our own internal vision system, note the disparity quantity that can be registered neurologically. So the disparity is registered, and we can make a good estimate of the distance of that point because we know that the distance is inversely proportional to the disparity. So far so good.

[16:54] Now, as I say, this is a very abstract diagram, and there's a lot of things we're leaving out of it, both for good and ill. In some ways, we can do a lot better than this, even in simple ways. For example, our eyes don't uniquely point forward. They can focus on a point. They can swivel. So that gives us more information depending on how much our eyes have to turn inwards to focus on a point using the two eyes. We haven't gone into that. We'll just use disparity as an approximation. But notice that given the structure of our eyes, we can even do better than that. Let's go back to the previous slide as well. So notice that we're able to compute something fairly good just from two parallel eyes. That is eyes facing forward, but we can even do a little better using slightly more complex computation. Also, I should say that this diagram

takes into account that the resolution on our retina is not constant. We'd like to, if we really want to focus on a particular point out in space, we'd like that point to project not just anywhere on the retina, but on an area toward the center of the retina, the fovea, which has high resolution. But we won't get into all those details now.

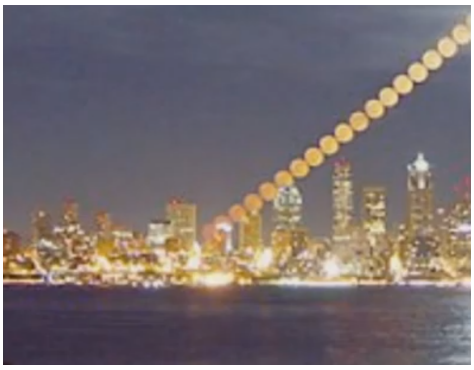
[18:41] If you've ever played with one of those stereoscopic toys where you can hold up stereo photographs to your two eyes and see objects at different distances and so forth, that basically uses this principle of binocular vision. Objects that are meant to look close to you in those stereo photographs have a large disparity. They have a big gap between where they project on the left and right retina. Objects which are further away from you have a smaller disparity. So if you were to look at the objects first with the left lens and then with the right lens, you'd see a bigger jump for objects which have a large disparity than you would for objects that have a small disparity.

[19:44] Now, there are some things about this. Let's go back to that previous diagram. Sorry I keep jumping back and forth. But when you look at that third equation, the distance is a constant over the disparity. The way that you might want to think about this, especially as a computer scientist or an engineer, is okay, that's a pretty useful formula for telling me how far a point is from the plane of the eye. Where might this formula be challenged? What might be the limitations of it? What might be the limitations of implementing it in a mechanical/computational system? How useful in practice is this equation? One thing that you might note off the bat is we're saying that distance is inversely proportional to disparity. Well, in general, as a heuristic, any time you see that the quantity of interest is inversely proportional to some other quantity, you start to wonder what happens when things get kind of nasty and that denominator on the right side goes to zero. After all, we only have a certain resolution in our retina. So there's a certain minimum size in practice that the disparity can get. What if it gets really close to zero? Then we're dealing with objects at a very large distance. What you can say in that situation, given a certain finite resolution of the retina, what you can say is that this equation is going to be much more helpful for us with objects at a short or moderate distance, objects at a moderate distance from ourselves, from the plane of our eyes. When objects get very, very far, we can't really use binocular vision to determine how far they are. Sometimes, when objects get really, really far, this kind of algorithm becomes effectively useless.

[22:19] So an example of this is looking at the moon. The moon is so far away from us that it's hopeless to be able to use binocular vision in our eyes to determine how far the moon is. The moon can't take the role of that blue dot in the diagram that we were looking at before. It's just so far that the D is so large that the disparity involved is extraordinarily tiny and goes way beyond the resolution of our retina. Which is perhaps one of the reasons that we sometimes have difficulty judging things like the size of the moon. In fact, I don't know that this is entirely related to this question, but I think it's partially related to this question. There's a famous question in human vision, you can call it an optical illusion. It's called the moon illusion, which is that we view the moon as being much larger when it's close to the horizon than when it's way over our head at the center of the sky. There are a number of explanations. No one of them may be the single explanation, but there may be several. I have a couple of ideas about this myself. I won't go into it, but for our purposes, if you look at that photo, it looks like the moon is huge. In

general, we have a very deep belief that the moon on the horizon is a much bigger object or it looks much bigger than it is when it's at the center of the sky. That's an illusion. The moon takes up the same real estate on our retina whether it's on the horizon, or whether it's at the center of the sky. But it sparks some questions, some curiosity about why this illusion would happen in the first place.

[24:41] One possibility is that we're using heuristics like, well, we see the moon next to a building like in this photo. We know that buildings are very large. They are huge. The moon is right next to it, and it looks like it's of a comparable size to that building. So it looks like it's a huge object perhaps. Since when it's at the center of the sky, there's no nearby large objects to compare it to, it looks comparatively smaller. That's one possibility. As I say, there are others, and people have come up with a number of different explanations for this, and they may in fact be complimentary. I remember once in graduate school because I was interested in science education, and one student came up to me and asked, "Why is the moon bigger on the horizon than it is at the center of the sky?" I hadn't thought about this illusion at that time. So they asked me that question, and I was just like, yeah, so why is it bigger on the horizon than it is at the center of the sky? I started writing down sketches and suggesting some equations and things like that, and after, I don't know maybe it was 20 or 30 minutes, I came back and said, it can't be bigger. I mean, it can't. It just isn't bigger. I don't see how it could look bigger. It's weird, it does look bigger, but in fact it really isn't. Take a look at this time-lapse photo. So there's the moon photographed at different positions over the course of one evening. You see it rising in the sky: same size. So it doesn't change size, but we definitely view it as changing size as it moves upward in the sky. Now, that could speak to one limitation of this binocular vision.



[27:00] Perhaps a more interesting question though has to do with, let's go back. I'm going to go back all the way to this diagram. There is an additional simplification of this diagram that we haven't really focused on yet, and that is this whole discussion has focused on determining the distance of a particular element, a blue dot, from the plane of our eyes. That was the way we posed the problem. In order to solve that problem, we noted where the blue dot projected onto the left retina, and where the blue dot projected onto the right retina. And we used those two corresponding projections to determine the distance of the blue dot, to determine the distance of the object. All well and good. Life and the world outside us, however, does not consist of one hovering blue dot sitting at a distance in front of us. It consists of thousands and thousands of visual elements that are projecting simultaneously on the left and right retina. So a crucial problem here, now we have to start thinking you know, again, we've

been caught, this course is called minds and machines. Think about what it would mean as a computer programmer, to try to make use of this formula. The crucial question which we've swept under the rug until now is what's called the correspondence problem. You have a thousand visual elements projecting onto the left retina, a thousand visual elements projecting onto the right retina, how do you match up the left retina projections with the right retina projections to make use of this formula? Notice that in order to make use of this formula, you have to know which projected point on the left retina corresponds to which projected point on the right retina. If you can't make that correspondence, if you have a thousand elements here on the left retina and a thousand elements here on the right retina, and you don't know which corresponds to which, you could come up with all different estimates of where objects are out in the universe.

[29:48] So here's an example of what I'm talking about, an experimental example. This is called a random-dot stereogram. It's a fascinating psychological experiment. The idea is these are very, they're not identical, but they're very similar patterns of black dots on a white background. If you look at these two squares with a plane of something blocking between your two eyes, or you look at it through one of those stereoscopic toys, what you will see, it may take a little bit, but what you'll see is you'll be looking at these two pictures and suddenly a square, a little and a smaller square toward the center of the two pictures, pops out at you. The reason for that is that a square region within the two different grids here, a little square region, has been offset with a particular disparity that is smaller, excuse me, the particular disparity of these squares is larger than the disparity of the points surrounding them. Since the disparity is larger, we see that little square region is popping out toward us. You can look up random-dot stereograms on the web. You can try them. Put some page or something in between your two eyes, so you can only look at the two pictures with two separate eyes. What you'll see is that you are able to discern a square region that's popping out toward you. How are you doing that? See, that's a really interesting question. In order to do that, in order to achieve that, you have to determine that the black point X on the left retina corresponds to black point X' on the right retina, and you have to do that correctly for all of the points in these two grids. That seems like a huge computational problem. It is a challenging computational problem, but it's one that our visual systems are able to solve.

[32:27] One of the readings that I am assigning, actually, as accompanying this talk, one of the readings talks about an algorithmic or computational approach to understanding how it is that we can solve the correspondence problem relatively fast. Again, we're talking about computational issues here. We don't just want to be able to solve this correspondence problem eventually. I mean, we don't want to be able to solve it after three days of staring at these two grids, we want to be able to solve it hopefully as quickly as possible, maybe within a matter of seconds. Why would that be important to us? Why would it be important to us to be able to look at highly complex scenes like this and to see the distances of objects within them?

[33:27] This here is a suggestion of why that kind of thing could be interesting to us. So this is a painting called *Pintos* by the artist Bev Doolittle. Notice that if you were out in the wild actually seeing a scene like this, what would you be seeing? Your left and right eyes would be seeing what looked like more or less random dot stereograms. That is, you have a collection of

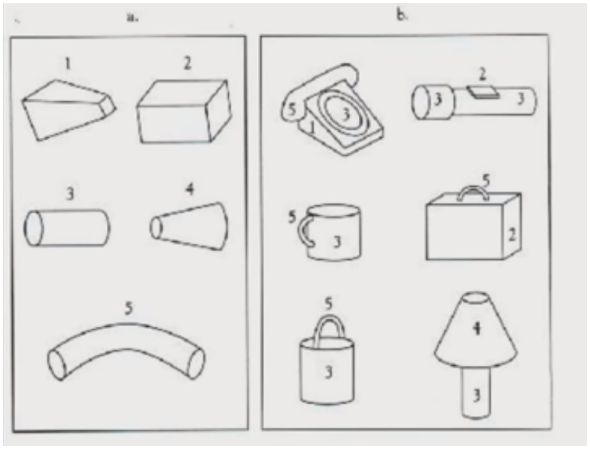
points and blotches of color on the left eye, points and blotches of color on the right eye, you're looking at the two of them. Some of those points are closer to you than the others and distinguish objects that are actually sitting closer to you. This could be rather important for you to know depending on the situation. I mean, these are pintos. What if they were predators? What if they were prey? You might be very interested in knowing where the objects are in this scene and in using your binocular vision to determine distances for them. So this is just one aspect of the vision problem. But again, like so much else in the vision problem, we make use of every trick at our disposal to get as much information as we can out of a visual scene.

3.4 Object Recognition

[00:00] What we're going to move on to for this discussion, is yet another aspect of vision as a computational problem, which is recognizing objects out in the world. When I say recognizing objects, I mean being able to look at the basic outlines of the objects and to tell what they are. An example that I sometimes like to use in my classes is to just draw something like this, and say what is that? What object is that? Most people, I'm assuming this might well be true of you as well, most people look at that and say, it's a hat. Now, think about that. Think about the task, the ability to recognize that little shape that I've drawn as a hat. There are a lot of respects in which it's that's a weird inference to draw. After all, the size of that thing is not especially the size of a hat. It's approximately a two-dimensional thing. It's a drawing. It's not really a hat. It's sitting there as a drawing on the screen. You have no color or texture information. You have no depth information. Why are those two little curved lines sufficient for you to be able to make an identification of this thing as an object? That's remarkable.

[01:47] We're going to look at one theory, one approach from the computational side of recognizing objects as part of what we do in vision. This is not the only theory out there. There are competitors. Moreover, this theory clearly cannot explain all the things that we do in recognizing objects. So it's only a partial theory, but it seems to be a very well thought out theory, and it gives us a good way of thinking about the computational requirement. How it is that we might program a machine to look at shapes like this and to recognize what they are. There are other ways, in fact, in practice, of doing object recognition in computer vision systems. But what we're going to talk about is one technique that seems to be part of or at least consistent with the human visual system.

[03:05] So this is called the Geon theory. The idea is that we recognize objects as collections of what are called geons. By the way, much of this theory is due to a researcher named Irving Biederman. So if you look up the geon theory, you'll see that it's closely associated with him, but he's not the only advocate of this theory. The idea is that, in a sense, we take objects, and we parse them. Much like you would parse a sentence into verb phrases and verbs and prepositions and prepositional phrases and so forth. We parse objects. We decompose them into linked collections of these geometric primitives. These geometric primitives are geons. Again, depending on the implementation of the theory there might be 30 of them. The actual number is not that germane to our particular discussion. Let's just say that there aren't a huge number of them. We're not talking about numbers in the thousands. We're simply talking about a modest number of geometric primitives that prominently include things like bricks, like number two in the left part of the diagram here. Or cylinders like number three in the left part of this. Or I guess there are different ways of



describing the shape number one. I think of it as a pyramid that has been truncated at the tip. But another way you could think of it is as a basic brick or rectangular parallelepiped. That is a polyhedron that's derived from a brick, but has a couple of non-parallel pairs of edges. Number four is a truncated cone. Number five is a bent cylinder. Again, you could add a bunch of these. They include things like cylinders with axes that curve or radii that expand or shrink, and they might include shrinking or expanding or curving at somewhat different qualitative levels. So we might, for example, distinguish a cylinder that curves only slightly from a cylinder that curves with a more dramatic turning radius. All of those things are, again, what we end up with is a collection of a modest number of these geometric primitives. Then we can put those primitives together in ways that can create simple representations of objects that we can then recognize.

[06:36] Take a look at some of the objects as represented on the right side of this diagram. Look in particular, let's look at the second row on the left, in the object section here. There's a cup which is composed of two geons: a cylinder like number three and a curved cylinder like number five. So the cup is a misrepresentation, it's a simple two geon object and we recognize it, as by parsing it into those two geons. The pail directly underneath it is likewise a two geon object, and it's composed of the same two geons, three and five: a cylinder and a curved cylinder. But the geons are positioned with respect to each other somewhat differently. So we're able to tell those two objects apart. Take a look at the flashlight up in the upper right of the diagram here. It has two cylinder geons: two number threes, and a small brick geon. So we parse the flashlight into two cylindrical geons of different sizes and a small brick geon.

[08:06] Many familiar objects can be interpreted this way, as collections of simple geometric primitives. Our job as a computational system in recognizing or identifying objects is to be able to look at an object, parse it into separate geons, and then essentially do a lookup, a dictionary lookup. That is, we look at that object at the top right. We see that it is two cylinders and a brick geon arranged with the two cylinders right next to one another, left and right, one of the cylinders is somewhat larger than the other. Then there is a brick geon that is positioned on the longer cylinder with the smaller radius. So we look up that representation, and we say it looks like a flashlight. Now, what are the advantages? Why would someone come up with a theory like this? Why would we go about recognizing objects in this way? There are a couple of standard ways of explaining why this could well be a good idea for a computational view of object recognition.

[09:42] First of all, let's consider this notion of geons themselves. So look at those shapes on the left side of this diagram here. The idea is these shapes are the kind of thing that you can recognize qualitatively from a variety of viewpoints and even with a certain amount of visual noise or with things partially blocking them. The reason is that they're defined in the kinds of qualitative terms that don't crucially depend on things like vantage point. Here's a diagram that explains what I'm talking about. Look at this brick geon at the top. What characterizes, what are the visual features that characterize a brick geon? A couple of the characteristics are that you have several lines that are parallel to each other and that appear parallel on the retina. That is, when you look at a brick geon, you see several lines in it that look parallel, and we interpret them as being parallel. In general, when you're looking at an object, if that object contains two lines that appear to be parallel, it's a very good bet that those lines are, in fact, parallel.

Sometimes we can be fooled, but most of the time that's a good guess to make.

[11:26] Also notice that corner in the upper right of the brick geon, it's labeled a Y. That's called a Y corner, and this terminology is derived from other algorithms and computer vision. But look at that intersection of the three lines, and you'll notice that they make a shape that is like a Y with the upper branches of the Y at the top there. If you were to take, I think I can approximate this with a book. Think of this book as a brick geon. If you were to take this book and move it in a variety of directions, you don't move it too dramatically, but you move it within a fair range of angles, then all of these qualitative characteristics remain the same. When you take one of these bricks and move it a little bit, and again, not too dramatically because you don't want to flip it around, but you just move it within 10 degree angles one way or another, the Y junction continues to look like a Y junction, the parallel lines continue to look like parallel lines. Similarly, well, here's a pretty good approximation to the cylinder geon. So as I move this, the parallel lines, line one and two of the cylinder, continue to look parallel. So that doesn't depend on the vantage point, and yes, you do see what looks like a narrow Y junction at the side here.

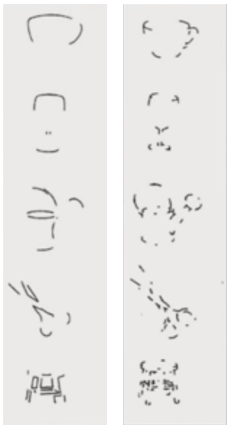
[13:13] So qualitative aspects of these geons remain the same, or the term that you sometimes hear is that they are robust with respect to viewing angle, and to a certain degree we can identify these geons even with a fair amount of occlusion. So it makes sense that we would build our geometric vocabulary using these robust, these common, and relatively viewpoint insensitive geometric primitives. So when we look at objects what we're trying to do is identify the geometric primitives in them, which leads us to a second question. When you look at an object, how can you determine where the distinct geons are? I mean, after all, let's go back a slide here. Look at the flashlight there. We very quickly can see it as three geons: two cylinders and a brick. How do we do that? How do we actually pass this object visually into several different geons?

[14:39] A standard idea is that we do this by looking at the joints of the object. A different way of phrasing this is concavities, that is concave regions of the outline of the object. Or if you want to think of this in computational terms, imagine that you were a little bug walking around the outline of this entire shape in a counterclockwise fashion. So you're a bug traversing the outline of this drawing in a counterclockwise fashion. What you are interested in, what these joints are, is the places where as a bug you're turning right. So if you think about it, if you're a bug, you're turning left all the way,across the bottom you're going straight, you turn left to go up the side, you turn left to go on the top edge, and then you turn right to go on that protrusion up there. So joints are the places where the bug turns right, that's one way of putting it. Now, is there some laboratory evidence that we use this kind of thing? After all, maybe we just recognize objects by, for example, if this were true, let's put it this way - if this theory is true, then we should see evidence that what matters to us in parsing an object into geons is locating the concavities or the joints of the object.

[16:27] Let's take a look at an experimental example that suggests that this may be true. Here are two views of an object partially occluded, top and bottom. So you're seeing the same object that is partially occluded by this, think of it as a black screen with holes in it. So you're seeing it at the top, and you're seeing it at the bottom. Typically, I don't know that experimentally this is a 100 percent, but you can imagine that a large majority of people find it much easier to

recognize the object in its bottom representation than in its top representation. Why? Because the bottom representation allows you to see the joints of the overall object, which makes it much easier to parse the object. If you are just looking at the drawing at the top, you might have a much harder time identifying what this object is because you can't see the joints as well. I should also point out that in both these drawings, in both these views, you're seeing if you want to think of it in these terms, you're seeing as much of the outline as in the other case. In other words, the amount of ink used to draw lines is the same in the top view of the object as in the bottom view of the object. So it's not like you're seeing more of the outline in one view or the other quantitatively. But qualitatively you're getting a much better view of the object in the bottom picture, because you're seeing where the joints are. So you're able to parse the object.

[18:20] Here's another example for you. So take a look at this collection of five objects, drawn from top to bottom. Now again, these are sketches of an object. They use a certain amount of ink to draw a part of the object. You could pause the video, whenever, take your time, but look at these five things, and see if you can make a guess as to what they are, what they're sketches of. People are actually pretty good at this, but they are better when they see, again, versions of these sketches in which the highlighted areas include the joints to the object. So now look at the second column and you have a much better view of and a much better chance at identifying the object. Because, again, even though the ink used in the left view and the right view of the object is the same, the amount of ink used is the same, nonetheless, the right view of the object gives you these joints, these concavities that enable you to parse the object into geons. Here's what the objects are in case you're curious. In class, I always have a certain amount of difficulty with that third object because I'm from Manhattan and so I always say it's a garden thing, it's a watering, it's a sprinkle can or whatever. But anyway, you know what these objects are. This is again at least very good circumstantial evidence that the geon theory is reasonable; it's plausible, holds water.



[20:16] Now, let's think about this again from a computational standpoint. What do we want out of a system like this? What do we want out of an object recognition system? We would like it to be fairly fast. We don't want it to take forever, in particular because every now and then we may be looking at something that could be very important to us or dangerous or interesting, and we don't want to lose the moment. So we want an object recognition system to be very fast. We would like it to work from an evolutionary standpoint. We would like it to work in dim light so in particular we shouldn't really have to make extensive use of color for much of our object recognition system. Color can help when we have bright light. I mean it's not something that we don't want at all, but our object recognition system should not crucially depend on well-lit conditions, which is why we're so much more concerned with outlines and shapes here than we are with colors and textures. We'd like it not to make too many mistakes. So think about the cup and the pail beforehand. They both consist of the same pairs of geons, but in different orientation. We don't want to mistake a cup for a pail, and vice versa. It just causes trouble if we make a lot of mistakes in identifying objects. So from a computational standpoint, we're placing some pretty strong constraints on this object recognition system. We want it to work under

non-ideal conditions. Maybe the object is partially occluded, maybe it's kind of dim light so we don't see color. So we want to work under non-ideal conditions. We want it to work regardless of at least modest degrees of changes in vantage point, in viewpoint, and we don't want to get things mixed up. We don't want to mistake one object for another.

[22:58] So now we might approach the question like this: how many objects do you think you have to recognize? We'll say in the course of a lifetime. How many distinct, and I don't mean like in this case distinct cups like if there are 10 different cups can you tell the difference between them. We'll get to that in a minute. But, think of the kinds of sketches that might show up on the right side of this diagram here. How many such distinct sketches do you think you should be able to recognize? You have a cup, a glass, a watering can, a scissors, a stool, a briefcase, a pail, a telephone, a flashlight. How many such sketches do you think there are that you might have to recognize? That's an interesting question. Do you think it's in the trillions, in the billions, in the hundreds? Well, a reasonable first-order guess is to say that, I think it's probably maybe in the thousands or tens of thousands. The reason that's a reasonable first-order guess is because our vocabulary is on the order of tens of thousands of words, our speaking vocabulary is maybe somewhere between 20,000 and 50,000 words, something like that depending on how you count. So since our speaking vocabulary is in the tens of thousands of words, and a reasonable number of those words consist of nouns like cup, glass, scissors etcetera, it makes sense that in general we're going to need to recognize or distinguish on the order of tens of thousands of sketches of objects.

[24:59] How are we going to store those? This is an interesting computational question. So you would want to store, somehow internally, the idea of a pail as being a 2 geon object with the two geons in a certain kind of orientation. I won't go into the details here. But if you think about this, you try to do some back of the envelope calculations. Let's say there are 30 geons, 30 distinct geons, and each of those geons could have three sizes. So you can have a large, medium and small cylinder, and there are, let's say, 10 choices of orientation between the 2, on the side, on the top, next to one another. I'm just winging it here, but the reason I'm just making these estimates are, 30 geons, each of which can come in three sizes, and they can be related by 10 different orientations, and let's just stop there. How many distinct two geon objects like the pail and the cup, how many are there? Well, each geon can be one of 30 types and one of three sizes. So there are 90 possibilities for geon number one. There are 90 possibilities for geon number two. That's by my calculations about 8,000 right there, and 10 orientations that these two geon could be in. So that's 80,000 objects that you could recognize. 80,000 two geon objects, in principle, that you can recognize. Naturally, when we get to three or four geon objects, the number grows hugely. So if you allow us to be able to recognize objects consisting of say three or four geons, the number of potential sketches that we would be able to recognize, if you were to map out all the different possibilities of geons and their mutual arrangement in pairs and in trios, and in sets of four, if you were to draw out that catalog, you'd have a dictionary of, I can't do it in my head - I'm sure it's in the many billions of objects. In other words, way more objects than we would ever need to recognize in life. Our dictionary space is much much bigger than our actual need for recognition space. What that suggests in principle is that the 20,000 or 30,000 objects that we really need to recognize are scattered sparsely in this

abstract dictionary space. The dictionary space is not full-up. It has space for, who knows, billions of objects, and we only have to fill that dictionary space with about 30,000 distinct possibilities, which suggests that we're not going to make too many mistakes. Any particular object that we recognize as a trio of geons or a quartet of geons, any particular object that we recognize is not going to be all that close in this dictionary space to other objects. We're not going to make that many errors.

[29:02] So if you view it in those computational terms, this is a pretty useful theory. Moreover, for many, many objects, we can make a pretty good guess at what the object is without even recognizing all the geons. So people would be pretty good at recognizing, if you look at that top row, you could probably guess that that's a penguin even from the first sketch. The second would be a little more informative, but you don't need too much detail to recognize an object. Similarly with the glass and the flashlight here, and if you recall, similarly with the hat here. I just showed essentially two geons, suggestions of two geon and a line drawing here. That was enough for you to recognize the object. I didn't need color, texture, many more geons or anything like that for you to recognize what this object was. So this is a pretty good object recognition device.

[30:08] Again, from the computational standpoint, what are the limitations of this idea? One of them is that the techniques by which we recognize geons are not fool-proof. In particular, we do things like we assume that lines that appear parallel to us are, in fact, parallel in the real world. Lines that look like they're continuous, but occluded in a certain region are, as far as we're concerned, continuous lines. Typically, those are good assumptions, but they can be fooled. So here's a lovely visual illusion from the photographer Walter Wick, the impossible doghouse. Look at this object. It should be puzzling. It doesn't quite make sense. How could an object like this be built? What does it really correspond to? This is what it really corresponds to: this is a view of the same object from a slightly different vantage point. But notice that in this viewpoint, for example, when lines appear to be part of the same continuous structure, but just occluded by something, we interpret them that way. We interpret them as the same object that happens to be occluded by something in the middle. Many wonderful magic tricks, as well as illusions, depend on fooling the visual system in this way. So the assumptions that go into our recognizing geons are pretty good. I mean evolutionarily speaking, they're pretty good. But they're not immune to error, and particularly, if people or other entities go out of their way to fool your vision system, it can be fooled.

[32:19] Moreover, there are features of vision and object, if you want to call it object recognition, that geons don't explain very well. Here are a few of them. When we look at a face like the face in the upper left there, that is Franklin Delano Roosevelt. So that's FDR, we recognize that face. From the history books, we recognize that face. Geons can't really explain that because from a geon standpoint, all faces are pretty much the same collection of geons. In other words, geons might help us recognize that that is a human face, but beyond that, it's of no use in determining who's face that is. So in the human brain, we in fact have special machinery. We have a special area of the brain that seems to be deeply implicated in facial recognition, and it goes well beyond the kind of processing that geons are able to do. Remember we were talking about cups before. On the upper right there, you could use the geon theory to explain that that's

a cup, and we could parse that object into geons and say that's a cup. On the other hand, suppose you were to start to pick up that cup and I say, "No, no, no, no, that's my cup. The reason I know it's my cup is because of that decoration on the side." That again, geons are of no use to that. So things like color and texture recognition are quite important once we get beyond the level of basic object recognition. Not only can we recognize an object is a cup, we can distinguish large numbers of cups from each other depending on their decoration, coloring and other kinds of things. So geons are not sufficient for determining individual instances of particular object types.

[34:41] You may have noticed that many of the test objects in the geon experiments, not all of them, but many of them are human-made artifacts: briefcases, and lamps, and flashlights, and pails, and watering cans. The natural world includes many sorts of shapes that are not that easily or profitably parsed into geons. Take a tree, for example. You don't really think of that as being a collection of, you could maybe at a very loose approximation think of that as a collection of geometric primitives. When kids draw trees, they often draw them as geon structures like a lollipop, a stem, and a circle for the tree part itself. So to a first approximation, you might even see the tree as parsed into geons, but that's not a very accurate way of looking at the tree. Again, you want a lot more detail. Many of the shapes in the natural world have these complex, sometimes fractal geometric qualities that render them sort of unsuitable for treatment by geons. So although it's a very powerful and useful theory for understanding object recognition, it can't and isn't supposed to cover all aspects of the object recognition problem.