Yes, you need to fix this check and what's going on with this one. Maybe the next person. Good. There. This should be good. Okay, I think maybe some of you still ask me to the bar. I did not download the bio. But anyway, you guys already see the correct answers. So please make sure you also a graph. Okay. You have any questions.

As this starts our today's lecture. Today we want to talk about the action. Last week we talked about this kind of filtering is a like intuitive, you want to do and we talked about this. operations by the correlation some pollution was there was some pillows, filters, linear filters. And we're talking about how to remove the noise. In today's lecture, we will learn about edges. And the place that I want any detection. So the direct you are images and some examples are directed and that's what we're talking about some grading of mass like a point down like this. So we are operated from complete the cycle and effects on 90. Next we will talk about the mark. Hedges or LOT detector. Lastly, we will talk about the mechanism and detection. Talk about the party talk about non maxima suppression and then we will talk about the HAC thresholding. First, this is about the chapter for this lecture. If you guys want to learn more, you can check it out. So now the key question is why edges. Are important. Any idea. It should the outlines. Yes. Why edges are important. Because it's very like key features.

Yes, it contains key features. Correct. So let's try to see some examples here. For example, this is about a car. About this. Yes. Okay, there right. What about this. Yes. So why you guys can tell the name of those kind of features. So that's just the best on those edges. Right. So this is all those images are just edges and there is no color information. That's about edges. But still, we like a human. Human beings have the ability to understand the information converter only by edges. So from now you want to say the edges are really important. And even for many, many models, they are able to detect the object just based on edges. They should be still very robust. Next, the formation of edges. So the edge detection is one of the most important pre-processing step for various computer vision tasks. For example, the object detection, some object recognition, some shape analysis, etc. To isolate an object from its background and never building object pixels, our first step is try to recognize its heads. So our edges are controlled across which the images of brightness or hue changes abruptly. So this edges are more than as a mathematical discontinuities. So there are several different continuities. For example, we have this surface mount discontinuities like the top part and compare with the side. And we also have depth continuities. That means the size of the object, the depth, is different.

The surface reflectance or reflective discontinuities is about the ink in this part. And also we have the illumination discontinuities. For example, we have the shadow in this image right so that you can see the difference between them. Here is another example about all those different kind of discontinuities. For example, we have this reflective discontinuity and we have this depth continuity about the mouth part of the bird. And we have the surface mount discontinuity and also we have this illumination discontinuities here. So now I said what is our aim here? What is our aim? Yes. So our aim here is very simple. Let's try to detect the head of this bird. This is the object we want to get. So given an illumination like this, we can directly tell whether it is a bird or not. For the neural network, what it is doing is trying to pick all those important areas in a bunch of small patches. For example, 20 by 20 by 20, you should be able to block all these 20 by 20 pixels. Then you have another 20 pitches. For a neural network, it is really want to detect so many small pitches of this big image. But again, for the neural network, it aims to find the features like those kind of edge information about the object. So they are eventually make a classification based on those pitches. The next natural question is how to detect edges? Basically, in the early stages, there are two easy methods of edge detection. There are thresholding and convolution. So I will cover these two one by one. So first, let's see the details of the edge detection.

So we will talk about the directives. The question is now how to figure out the pixel on location X, Y is an edge pixel? How can tell that pixel is an edge? Change like in neighboring pixels. That's fine. So now the question again is about how to figure out the pixel on the location X, Y is an edge pixel. Previous thing about our initial discussion of image as a function. So this image has a function of I X Y. And there are two representations of the same signal. One is about image, another one is about height map. So edges are clearly visible in the height map. Where there is an edge, there is a still clear impact in enhancement. So that's the change from the height map. So now we can see more, right? I think most of you, even like in this way, can see the difference of the intensity here. As some of you guys said that we can tell it's a neighborhood, right? To see whether there is a bright change there. The best idea is trying to look for a neighborhood with strong sign of change. So now you see the significant changes here. You see it? But another kind of question is we may need to care about what is neighborhood size, which means that how many neighbors you need to be looking for, right? And how to detect changes. So how big of a sign should we call it changes?

So as you can see in this example, there is a change in the intensity of light. So the 80s are around one side and the 20s are on the other side. Can you say that is a significant change? Let me ask. Yes. Okay, some of you say yes, but some of you may say no. Because sometimes the stairway depends on the neighbors, right? This can be a little bit of a change of the intensity. So now let's think more about the changes of the intensity. So the mh is a place of rapid change in the image intensity function. So changing the function is detected from the irreptives. So by now we change it to another question that is equivalent to finding peaks in the irreptives. So here is an image. Suppose we have this kind of image and let's try to draw a line. And here is about the intensity function. So around this horizontal line, so first it should be bright, it should be a high number. Eventually it will back, it will become very smaller. And then it will go back to here.

So this is about this line. Let's say if we want to apply the irreptives to find changes in this function, what can we get? If we apply the irreptives to here, what will we change here? No? No change? What about this part? No idea? It becomes smaller or becomes bigger or becomes smaller. What about others? What will we get if we take the irreptives out of this? No idea? Let's see what's going on with it. So really, especially in this part, the edges are corresponding to the actual amount of the irreptives. Because this is like, it's more like calculating the slope of it. So now the slope is going down, going down here. However, if you want to go to this part, your slope should be going up. So that's the reason it's increased here. It's about the first irreptive. Let's try to see more details of it. So let's say this part is very about a negative slope. And this part is about the positive slope. So this is the corresponding between your first down and this is the up part. Is that clear? Yes or no?

So now you will notice that if we are able to find its first irreptive, we will tell whether it is an edge or not. Right. Because the significance in these two places, whether it is an edge, whether it is small, whether it is very big, once you see this kind of big change, you will say, oh, there can be some edges there. Let's try to review this simple calculus of it. So let's say we have a function f of a that is equal to a. If a is equal to 1, then f of a should be equal to 2, right? If a is equal to 1.4, then f of a should be equal to 2.8. And the slope is calculated by the dx divided by dx of f of a. That is true, right? So basically it's just about 5 divided by these points. 0.8 divided by 0.4, then you get a slope of 2. But no problem. So next question is how to find the image directives. The same way as we did to remove the noisy. For example, we need to run some filters here. We need a filter or the differential operator. The differential operator is applied to an image and then you will retain its directives or so-called grading here. And those operators can be modeled as filters, masks or kernels. In the last, some strategies apply to grading to select the h pixels. And what is the grading? So that is about multivariate calculus. For multivariate, which means that a function with more than one variable.

So the image is a function x and y. So in this case, we call it as a gradient. And for signals with more than one variable, we can take the partial directives. For images, the derivative in the x direction and the directives will be distributed in the y direction. We have two directions here, right? For the grading, it is a vector map of those directives. So it's basically combined with x and y. And here are the details of the gradients in the image. As you can see there, it consists of the partial derivative of f over the x, the partial derivative of f with respect to y. Do you guys have questions about those? No? Great. Then, an image gradient is a directional change in the intensity of color in an image. Mathematically, the gradient of an image at each pixel is a 2D vector. And with the components given by the directives in the horizontal or in the vertical directions, the gradient points in the direction of most rapid increase in intensity. For example, in this case, if you see that the image changes in the x direction, this is the x direction. This way you can see the image is changing this way. And this is what we call the gradient of x, which means that this image is changing only in the x direction. Thus, we have this kind of equation. So if you have a gradient of x, you only have x with respect to x here. What about y? Does it go down? Yes, that's right. So if your gradient is only changing in the y direction, so you may need to move down. So then eventually you will only have a gradient that is with respect to y here. What about the gradient that in x, y, x and y direction? So in this case, you will see a significant difference between like in this way, right? Because now you see a ghost in x and y. Let's try to talk more about the gradient of an image. What about the gradient direction? What does this mean? What's this sign? Huh? Say again? Say the w. So actually this is the upper tangent of it, right? So we can say that the gradient of x is changing in y direction. So that's y in y direction. Plus the y in y direction divided by the x direction. So that you can get the direction of it. Or we can say that the change is changing in y over the change in the x direction. So we can calculate the gradient magnitude for this one.

This is just a part of the square root of it. And this is very important for the edges. So we will see a few examples about what are those changes in the x direction and what are the changes in the y direction and what about this gradient magnitude. So for our 2D function f x, y, the partial derivative that is defined in this equation. Let's say we want to calculate the partial derivative with respect to the x. Let's try to get a limit that is once x is close to zero. So this is how we define this partial derivative. And in the computer vision we are dealing with the disparate data or images. We can approximate using the finite difference, which means that the partial derivative with respect to x, that is almost equal to this equation. And this is part of the right derivative of the water difference. And in a similar way, we will calculate the finite water difference and the central difference. Is this a derivative of the finite difference? Is this in the x direction or in the y direction? Is it in x or in the y direction? X or y? Y direction. How many of you say it is in the y direction? Why? We have 1, 2, 3, why we have 1, 2, 3? So then I assume all the rest think this is the x direction.

Why I think it is the x direction? Because the x direction is the x direction. So let's try to see what's the reason of it. I think the majority of you get the correct answer. So actually in this kind of difference, those vertical lines have a high response or horizontally high response in the x direction. So the reason is that the x direction is the x direction. So now we have more tests. What about this? This is x or y. This is x or y. This is x or y. This is x or y. And this is y. This is x or y. This is x or y. And this is y. about this this is the X or Y this is X or Y you think this is the Y what about this one this is the X or Y let's try to see more this one and this one let's do a guess how many of you think this is the X this is the X how many of you think this is the X true then I think the majority of you think this is X really this is really X look at again to see to check which like your rectum this is Y this is Y this is the X so that's it also with all those so actually this is about the X direction majority of you guys did a better job in this case right and this one is very about the Y direction you see it kind of confusing so this is should be Y and it should be changed why is that? because in the X direction actually more like from this way you can see it carefully and for the Y majority from the top you can see some kind of difference here okay yeah this is Y and this is the X let me try to change it immediately so it will be a confusing idea okay I see maybe in the future you guys can know which one is right right what about the gradient magnitude? why can't I get it this way? can't remember what is the gradient magnitude? how we calculated this one?

yeah it is like a square root of the sum of X to X square plus the Y to Y square so now in this case you can see one of all is just an outline right and this is about the code and also something you guys need to replicate in your homework this is about read the images here and then calculate the direction in the X direction then calculate the direction of the Y direction and then eventually you guys need to plot out the magnitude so that is your homework here just directly about the answer you can get this X direction get the answer of Y direction and then get the answer of magnitude okay so you not get some kind of wrong answers but the answers are already here so next part we will talk about gradient mask specifically we will talk about the super operator it is required to have operator or kind of mask or kernel that can be applied to an image to implement this kind of partial derivative with respect to X here is this a good operator? is this a good operator or good kernel?

is this a good operator? any answer? this is good or bad? if the gradient is better then it is good you think it is good? the answer is what? why you think it is good? I think it is a good operator and it is a good operator and it is a good operator and it seems like one reason any other answer about whether this is a good operator? I just have one answer the kernel of this should be a square say that again it is a bad kernel because kernel of this is a square should be square? you mean like 2x2, 4x4? square? yeah 2x4, 4x4, 3x3 3x3 well, you have a close answer but not 100% correct remember we mentioned about the kernel size usually you get plus 1 is what number? is it even number? or it is an odd number? odd number, but now the key the correct answer for this one is no because it is not a good operator as there is no middle pixel in this column, which you can see those are good masks, you only have kxk, whereas k should be an odd number like 2 plus, that is just about 2 plus 1 remember in the future you guys should use some odd number as your kernel size okay, you get it? and then here it is about really about the supermassive that we want to discuss the super operator is a discrete differential operator that can compute an approximation of the gradient of the image intensity function so here about the x and the y direction so this is about the details of the supermassive mask another question is, why we want to scale it by 1 over 8? I will quickly guess why the total image of the operator should be 1 over 8 say the number? the total image of the operator should be 1 over 8 the total image of this one is 8? what? weight? you said the weight of this should be 8? yeah is that right? so let's say that you have, what's the reason here? for example, this is the response is about what? in the left hand you see what's going on this is about what? zero zero, right?

so that is minus 1 here so the pixel here should be zero, right? so that is minus 1 times with zero then zero times with zero and this one just one times one so eventually for this part you can get a number of four here this is one case, what about the other case? this is about another way, so this is one minus plus one times with one then zero by zero, then one times with zero so eventually you will get another negative four in this case once you try to across all the whole image and eventually this should be the key reason that is like the range between negative four and four then plus four, so eventually you will get zero between zero and eight and this is the reason why we get a scale right over eight this is really about the range of final results to compute the gradient vector as each pixel by moving from boring images with a horizontal and vertical deductive filters or masks compute the gradient magnitude as each pixel so magnitude as a pixel is bigger than some kind of variable we will try to report it as a pulse point, eight points so let's try to see more examples here so first we can compute the gradient vector as each pixel by comproving it with the image with a horizontal and vertical deductive mask first let's try to get this is about the x direction and this is about the y direction this changes this time maybe better to see it right this time is better, x and y so why you get this gradient in the x direction and gradient in the y direction you are able to calculate the gradient and this is eventually the gradient that we can get and in this case, let's say you the gradient is bigger than some variable we are treating it as a pixel and this is the gradient that we can get so the gradient is bigger than some variable we are treating it as an edge if not, we will just treat it as zero so

let's see what are the differences here and this is how it looks like if it is bigger than some variable, it should be like any point and here are the results of different variables about magnitude for example, this is the threshold of 50 here are a lot of edges and you can see many parts in between so once you try to increase the threshold values you will see less and less about edges this is about 100 this is about 150 this is about 200 edges and this is about 250, 300, 350, 400, and 450 you see all the changes here so this is getting messed up it is getting longer and longer it becomes like pure and pure or smaller and smaller because you have very high threshold but I think which one is best? Your mind? you want to use this one? the third one? the third one? so sometimes it is kind of tricky you need to pick out a good number of thresholds so that you can not only detect enough details of your figures but also you need to knowledge shoot too many details for example in the first case there are too many details, probably you do not get it and here is the zooming of that part this is about 105 as I said this probably can be better case than the others 150, 175 and here are the details of code how to produce these kind of figures and also you guys need to do it similarly, this part with those images and calculate the gradient in X and Y directions and then you can get the gradient magnitude calculated right so eventually we try to get different thresholds so that you can produce all those different images again the answer should be similar here next we also have other kind of masks for example we have pretty masks so this is in X direction, this is in Y direction and we have robots masks in X direction, Y direction although in this case it is still like

2x3 dimension as we said previously it is not a very good operator to have this kind of 2x2 right but instead can detect something but eventually compared to this kind of pretty mask this one is not do a very good job it is not that you cannot use magic signs F2 by 2 but the key thing is that we more care about the results part because from here the results are not good so we try to avoid this even numbers of your kernel size next we want to talk about the effects of noisy and also we will try to remove the smooths again so consider our image row our column of the image so proteins are tested as a function of X so here is about the function Fx, big X so we actually see lots of noises here right and then once we try to apply the derivative operator with respect to the X what is this? can you tell where is the edge? can you do that? can you find where is the edge? no right because you have so many noises it is very difficult to tell about where is the significant change it seems all of them changes a lot so in this case we will either try another technology or we will try to smooth all these functions as they are what are the details? so this is the first example about finite differences corresponding to noise and here is about their mean additive

Gaussian noise this is about several different cases and here is about another example this is the noise image you try to calculate the gradient you can also find some kind of noise in your gradient images and in this case it will make it much more difficult to detect where are the edges and here is another example once you put a lot of random noise and put the noise on the image even you have some thresholds like 50, 100, 150, 200, 250, 300 it is very difficult to see the edges so eventually once you have another very high value you can very detect where are the edges so previously you guys see that 150 already give you a kind of good result but now you see it is not that good because of the noise how to join with noise images? how we did this before? can you remember? applying the filter so which filter is the best? that is the medium so similar idea we will also try to add a filter to smooth the image then try to get rid of the noise in here this is about kernel that is mostly Gaussian kernel actually this is the kernel that we want to apply over all this image and eventually once we try to this is the control function with the Gaussian kernel that edge times with f this is f and this is f this is some Chinese theorem and this is the part once you try to apply the directive operator that is respective to the ideas so now you can really looking for the peak the peak will tell you what it is an edge right?

so now it is really about peak and this part should correspond to the edge and eventually although you have so many noises there you can have what you see this is really about the edge part you get it? in the noise image you can better apply the smooth function to be there so that you can calculate the first directive remember previously once we tried to put this the first directive with the original f of x which is very fuzzy right? you still cannot see where is the edge after you perform the smooth operation there you should have a good edge there so in order to find the edges we need to apply a smoothing filter before applying the directive operator so the linearity and the associative property of linear operators so that is to say the filtering and the directives are linear operations and this saves the operation this part you can donate in this way and this is the f and this is the directive that is our edge that is respect to x here and here is can be similar as before this is just some kind of operations you guys get it? how to remove the noises get the edges? got it right?

good next part we probably want to mention about the MARR address so called LOG edge detector maybe let's take about 5 minutes break so then we will cover the rest of the slides what happened? what happened? it's fine it's there but it's not working yeah it's fine it should be there it doesn't work it's fine it's wrong so here it's fine it's fine it's fine this is fine it's fine it's fine it's fine it seems fine it's fine it's all the same but it's fine it's fine some of them are very similar they are very similar you know you can check them sorry yes okay that's fine it's fine it's fine it's fine okay that's I'm too Okay. The more we talk about. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most interesting. The most funny. I don't know. I don't know. I don't know. I don't know. I don't know. I don't know. I don't know. I don't know. I don't know.

I don't know. I don't know. I don't know. I don't know. This is this is This is This is This is This is Over all, that's the logic is not correct. As you do I run them and it's well raising points it's not a good way isn't not good points. I will wait to do that you were so I wasn't outless by to take care of the me but I was not good points. So I was not good points. So I was not good points. So I was not good points. So I was not good points. I will wait to do that you were so I wasn't outless by to take care of the me but I was not good points. Why was it different. The sample should be the same. Your test is going to be the same. Yeah, like, it's the same. I'm doing on the sample. What do you mean, you shouldn't use this one. What's the way. No, it's probably a new part. It's just about. Believe me. Yeah. That's not theSo. Or the. Will you will. Win the weird. Once. And then.

Then. Yeah. I tell you, every moment. Because you come every book I will read. Okay. Probably we can. That's what I talk about this Mar. So here's about the overall history about this matter. So the actually it's a very tomorrow. Practitioner. The scientist. Researcher. He wrote about. So why he wouldn't work in MIT and took the inspiration from the human. Here So let's see the details of it. Smoothen an image by applying Gaussian filter, that is about apply to the S. Then we apply the vibration to S, that is to say the laplacian is the second derivative. And then we try to find zero crossing, which means that by applying first derivative we can check that is maximum or minimum of a function. Like we did in the SOVL edge detector, the second derivative of maximum or minimum will be zero. It solves the problem of finding correct spherical veils with the second derivative just to find the zero crossing. In this case right. So a scan along each row records an edge point and the location of zero crossing repeated for them for each column so that it will get zero. So here is the details.

So this is about the Gaussian smoothing by the S is equal to T the fans with I. Where S should be the most image and G should be the Gaussian kernel I the original image and the G here is the function. The laplacian of Gaussian is really about this is what it means what. The second derivative of the Gaussian prime that is respect to X and Y. Where this laxator can be a laplacian or this one can be just a gradient. So here is about some properties of the. Like, for example, in this operation, we're using this as a property. So the second derivative of S then the second derivative of X. Which means like second derivative of G times with I. So that means that we can find the second derivative of G here. And then we can find the second derivative of X. And then we can find the third derivative of X. And then we can find the fourth derivative of X. So that means that we can find the second derivative of G here. Is this correct? Is this correct? Okay, good.

Which means that we just need to find the second derivative of G here. So this is about second derivative right and the gradient of the G should be like this way. Right. Right. Right. Right. No matter how you guys will see it later and then you guys need to prove why this should be equal to here. And then here is a realization of the. Iog detector or the MA, hydrogen's ag detector. So it is like a magnet, max and hat. So this is about from the top view. What about from another view? Let's try to see the difference here. It's really about the head here. See the big change from top to the bottom part. And this is about the second derivative. And the results is about finding the inverted mappian hat or the lablation absorption operator. Now you see the changes. I'll apply to here. You can see this edge again. And actually there are four cases of zero crossing you need to monitor. For example, you have positive, you have negative. For example, one pixel is like positive and the next one is negative. And in this case, you will have zero in the middle and you have negative in the left and positive in the right. And you have this left in the left, there in the middle and positive in the right.

Then you need to find the half one as zero crossing. So by looking at the slope to recall each pixel, any pixel, the slope of A and minus A should be calculated by A plus B with A plus B. This is absolute value. So then we will try to apply the threshold to your slope again. So to say where is your slope. And we probably need to find a good threshold between these. So here is LOG again, but with a set of one. It's about all two. See the changes? See the changes right? This is about three. About 20 here. You see this is about 10. There are lots of threads. This is about try the threshold.

Yeah, more and more. You like this one? It's good. And then at least it can show lots of details of your image. Now here is about the code again. You need a grid image and then you can get the sigma and then you have the different kernels and you'll find the different threads. So that you can get the similar results as you see here. Next part we will talk about our last edge detection method that is called the K-measure. So before that we will talk about the polish of the edges. Let's say in the left hand, this is the true edge that we want to detect. Some methods like, it will give you some results. For example, in this case, it is very poor. It has poor robustness to noise like this case. And then it has kind of poor foundation because you cannot get a straight line on the sun. Edge detection methods that will give you too many responses to noise. And this should be several stages for Canny detection. For the Canny edge detection, it did a pretty good job in considering quality of edge. Referring to previous slides, we have so many different cases. It has been widely applied in various computer systems.

The K-me measures approach is more English or engineering oriented as opposed to L.O.G. which was motivated by a series of neuroscience. So there are several criteria. The first one is about the good detection. Then try to find the best way to detect the noise. And then the second one is about the high-quality. So this is the K-me measure. The first one is about the good detection. Then try to minimize the probability of force, positive and the force negative. The second criteria is to get a good localization. Then try to detect the edges that are closest as possible to the true base edges there. And your third criteria is try to get some single response between. Which means that the detector must return only one point for each edge point. You shouldn't give many more responses there. And the K-me proposed edge detection problem as a mathematical optimization problem. And there are several steps of Canny edge detection algorithms.

The first one is about applying motion filters to smooth the image in order to remove the noise. And the second one is about finding the intensity of the image. And the third one is to find the magnitude and orientation of the gradient. And then you need to apply the maximum pressure to get rid of the lots of responses to the edge detection. And eventually we will try to apply the heteresis threshold. So we will talk about this heteresis threshold later. So as compared to LOG, which uses the technical derivatives, the process with the steps, these three steps are very similar to the spurious operator as well. So Canny uses the first derivative. The second derivative is the heterogeneous threshold. So the heteresis threshold is the first derivative. It is known that it is expressed as non-maximum, which reduces the positive and negative, but it applies some heteresis threshold. This is about the contribution of this method. So let's talk about more details of what is non-maximum suppression here. So K-me suppresses the pixels in the absolute value of the gradient S, which are not local, maximum. So suppose this is the true edge.

So Canny’s method will not suppress an edge pixel based on local, never-informed information. Rather, it will look at the gradient direction or orientation, also called the direction of gradient, is perpendicular to this edge. You find the pattern in the neighborhood pixels. So we will see more details about it. So now in this example, let's say the pixel X, Y will be retained as an edge pixel. This will be treated as an edge pixel only if it's greater than X prime and Y prime. So the same direction of X and Y are the same direction of X and Y.

And this is the so-called cross-time edge detector using the direction of a gradient. The gradient will need to compare one point here and one point here. Here's another example. So in this case, we will try to check if the pixel is a local minima along the gradient direction. So here is the gradient direction. This is the value of P. This is the value of R. It requires interpolating the pixels P and R so that you can really tell whether this is the edge or not. So in this case, we will really have a half-bound and a lower bound. So in this case, we can tell whether this is the real edge or not. Previously, what we needed was that for example, in the so-called detector, we only take about this kind of second directive like this, whether the slope changes significantly.

So then we are treated as the edge. But in this case, we really needed to compare several other pixels here. So before non-max expression, you can see this is the results. So after you apply this non-max expression, you can see some significant changes here. So now in this image, you can see this kind of edge is really thick. It's thicker than what we want to get. But in this case, although a little bit darker, you can see this edge should be more like a real object. Because this is just too much. So next we will talk about the hysteresis threshold. So previously, studies study that detector used one threshold value to record a pixel as non-Edge. So Canny’s method uses two threshold values. The high is about a high, just a very high value, we call the pixels as as non-Edge. And low, if it is below, lower, so we call the pixels as non-Edge. And then what about the people between high and low, they will be recorded as an edge pixel. If it is connected to an edge pixel, it is due to the fact that the edges do not occur in isolation. So this is the case that we want to explain. It's hysteresis threshold, the y-axis should be the gradient magnitude.

So all the connected pixels, although below high threshold, we are going to call it as an edge pixel here. So the edge-connected pixels above here, up below this part here, we are not going to be recording as an edge pixel, as they are not connected to pixels above the highest threshold. So the majority of the time we will focus on the middle part of it. So now let's see what are the differences of the hysteresis threshold. So this is about an image. So if you use a very high threshold, this should be very strong edges. If you use a kind of low threshold, you can get some weak signals. But once you try to combine them together, you will get this kind of hysteresis threshold. So basically that combines the advantages of the high threshold and low threshold. And here should be about more details of the original images, how we calculated.

So first step is to fill the image with derivative of Gaussian. You guys already know how to do that, right? And next, find the magnitude and orientation of our gradient, then apply the non-veximal expression. So now we have the non-veximal expression. So lastly, you will think about the threshold that is really above heterogeneous systems. First of all, you need to define two thresholds, low and high. And then use the high threshold to start edge curves and use the low threshold to continue them. So this is all about this kind of detector. And let's try to see some results. So this is the first one. So this is all about this kind of detector. And let's try to see some results here. This is about the down part of the original image. And this part is about the Kd result between 0.01 and 0.03.

So this is about the difference of it. So here compared to previous results, you can see that in this case, especially about those parts, it didn't give you more details in the image, right? And then here is how we implemented this final Canny ED detection with the images. And this part is just about the Canny ED results. You need to define your kind of threshold. I believe it should be a majority of content of today. Maybe we can talk more about some kind of effects of edge detection. So actually now, even up to now, I would like to say even we have so many very powerful devices, edge detection is always like our kind of tasks. This is about the one result of previous human segmentation, like this segmentation result. So this is about the ready magnitude. Already you find lots of kind of significant features there, right?

So now it's the last one that you guys need to convert your MATLAB code to Python. And you need to prove the second derivative of the quotient. We'll put it in the canvas. And you guys can start to do it now. But in this case, I want you guys to finish one or two questions, then you can go. Because I feel that some of you did not catch the important part. During the course. But if you want to go, you definitely need to show me whether you finished the possible ones or not. Let me try to do it. You guys can have a look here. So let me know whether you guys can see. Homemarks are not. Oh, maybe I can expand the homemark. So once you guys open this one, you will see there should be five. Just the five points. Five. No, just the five classes here.

So first of all, you need to calculate the image grading of the type of image. You need to read and then choose the grading of X and Y respectively. And you guys need to implement this super edge detection of bird images. And then you guys need to implement the O.G. Edge detection. And this is about the kind of edge detection. And this should be about the proof of the second gradients. Some of you from the machine learning classes, you should get a similar kind of question before. So now you guys. So now you guys.