Advanced IOT Applications

Dr. T V Prabhakar
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

Lecture – 12
Basic computer vision algorithms Part -1

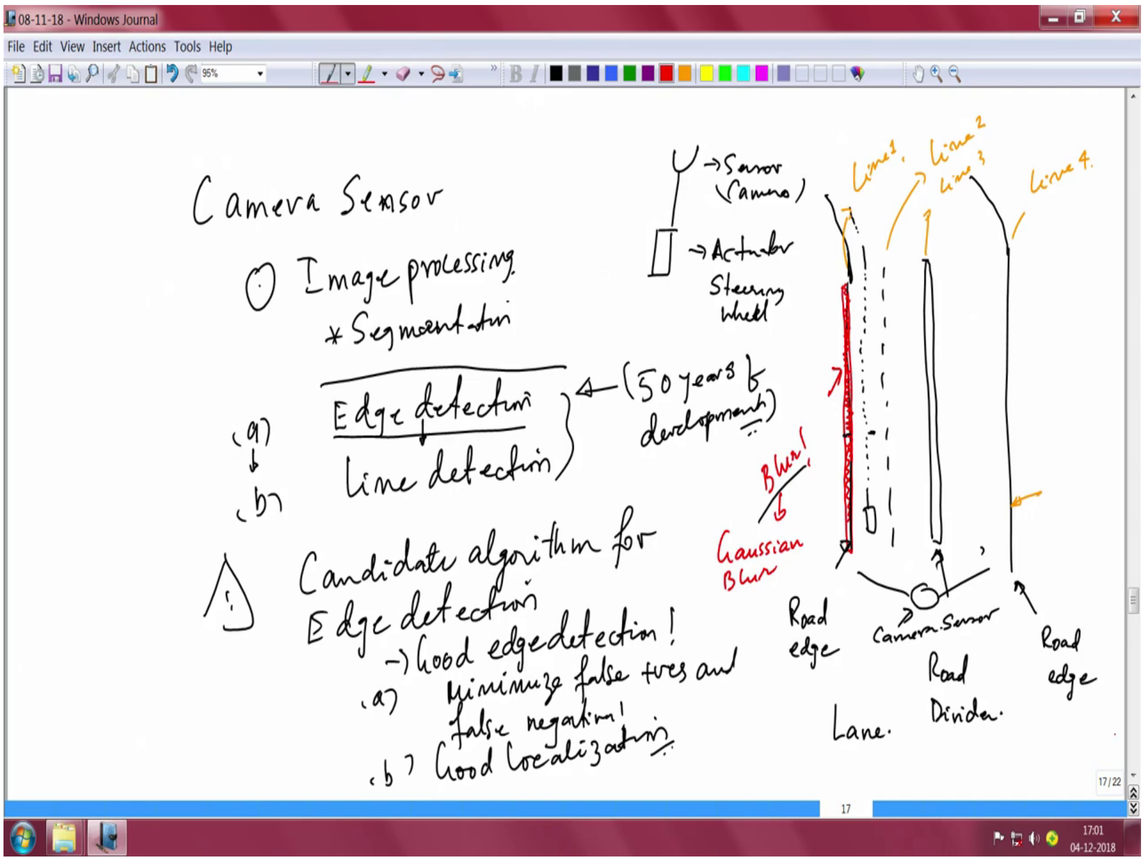
Let us now look at how we can apply camera sensor for our automotive applications. To see if it can be we can use this sensor for building an IOT application. As, you know 2 hot topics exist today in the world of automobiles IOT and automobiles; one is in the area related to electric vehicles. How IOT can assist the use of electric vehicles and what kind of technology can IOT offer is one question. The second question is about autonomous driving. Many courses are there on application of artificial intelligence and machine learning for automobile applications, and autonomous driving and all that.

So, it is a hot area. We cannot obviously deal everything here and our focus is really not there. Our focus is more on a very narrow part of all what we are talking; one thing is if you want to do any one of those you want build any one of those applications, you must know some real basics. And, what we will try and do is to cover those basics including your own ability to experiment in at least a simulation environment. That is the key and is what we want to achieve. So, let us take an example of a camera sensor. Now camera sensor obviously, gives you an image.

Apply this image and the series of image is which make a video at a particular rate when images change at a particular rate we get a video screen. See, whether this image can be used to detect important things, when a vehicle is moving on a road. Now, this itself is a hard problem. First of all if you take Indian conditions, how do you define what a road is? If you go to the slightly rural and village side there is no road at all. Now, what is a road, what is not a road is difficult to even identify right.

So, that is one part of the story. Assuming that you are not there you are in a urban situation you are in a large city or at least in a reasonably reasonable city where a lot of traffic out there, there are roads and there is a tar road or a concrete road, there is a road and then there is a road divider and all of that.

(Refer Slide Time: 03:18)



Camera Sensor

(i) Image processing

* Segmentation

(a) Edge detection

(b) Line detection

(50 years of development)

Candidate algorithm for Edge detection

→ Good edge detection!

(a) Minimize false +ves and false negatives!

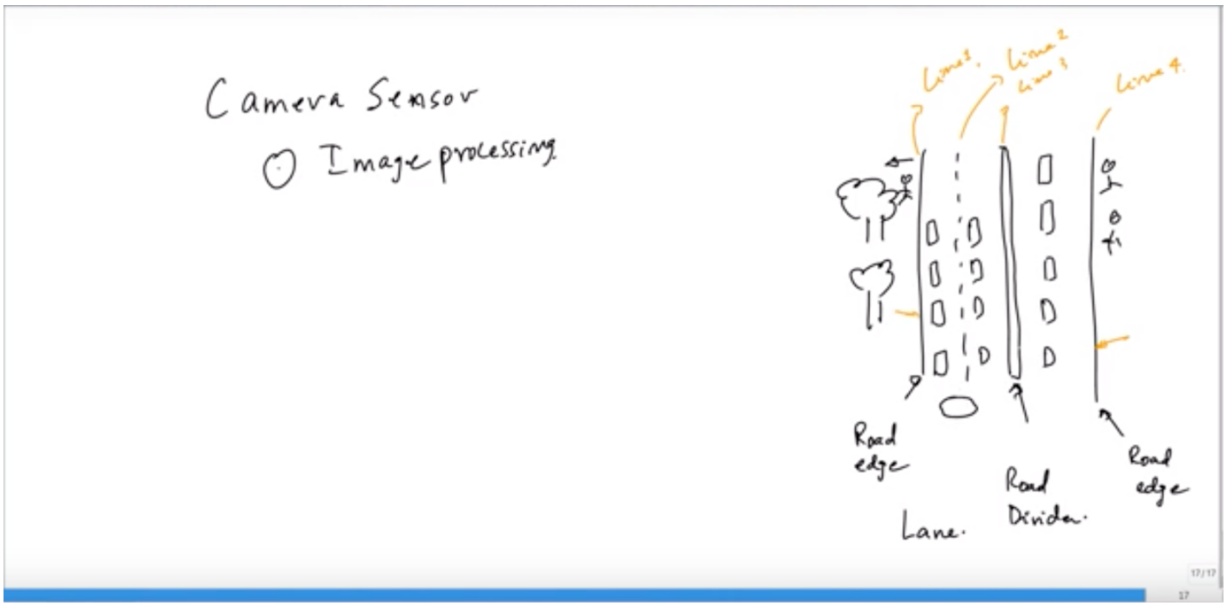
(b) Good localization

So, let me point you to this picture. You have the road edge on the left, you have the road edge on the right and road divider; perhaps a big monster in the middle, which is trying to sort of ensure the traffic moves in a disciplined manner. If, this is not there it can be chaotic because our traffic coming in opposite direction can actually come into this space and, traffic in this direction creates a complete jam.

So, they put a traffic divider. Not only that there are also lane separators, there is a line typically a line which is sort of line marking which actually say that this is a lane and this is another lane. So, you have lane 1 and you have lane 2. Typically one of the lane should be a fast lane; the other should be something for moving at a constant speed. So, things like that, but often you know vehicle just go on either side without really worrying much about what is a fast lane and so on.

Now, if you look at this picture, here you will see that this is a line which is something that you want to detect, ultimately you want to detect the line at the edge of road. How do you use a camera sensor and what kind of image processing tools are required to do this image processing to detect the road edge? See, you have to note one important point, this is something that should be derived from an image.

What will you see in an image typically for camera is kept outside the windshield, it is not seeing the nice picture anymore. It is seeing vehicles, it is seeing vehicles in this direction, it is seeing it is own vehicle, it is seeing some vehicles in the positive direction, it is seeing a tree trees, then it is seeing people, it is seeing people walking, it is seeing cyclists, it is seeing vehicles moving on either direction also in it is own direction.



So, if you put a camera sensor on centre of the left lane and assuming that it has a good field of view. It is getting a good field of view, you are seeing all of this, which means you have to break down this image into segments and perform a segmentation of the

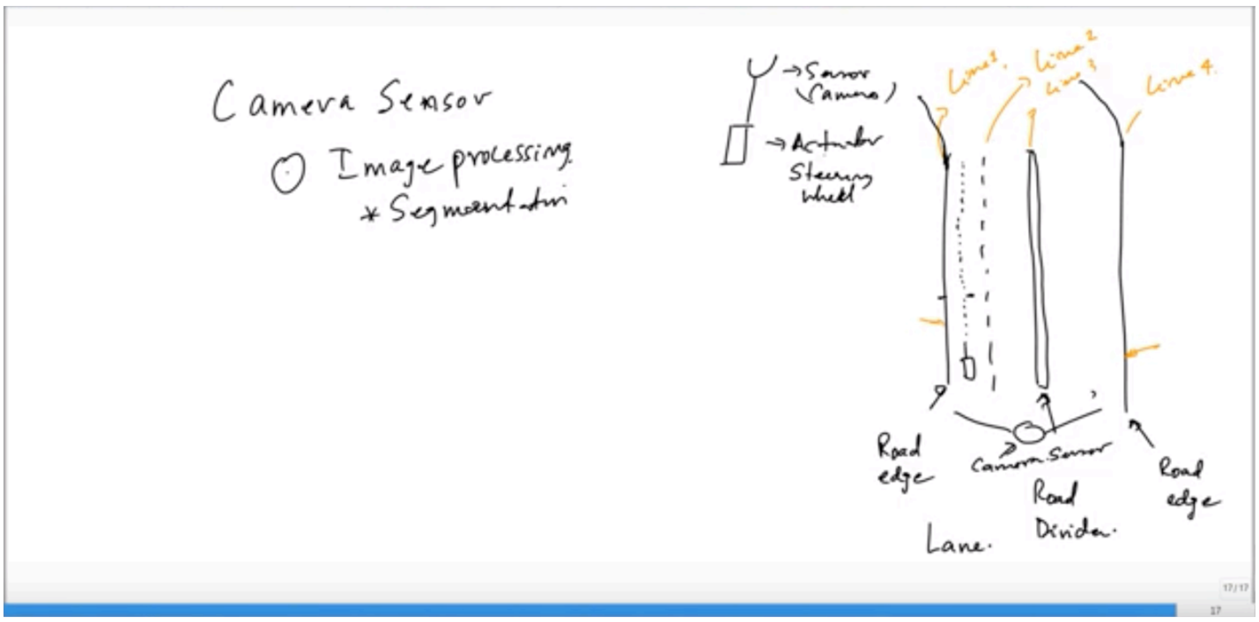


image and only detect and only detect two lines of its own lane. It's not so easy already if you want to get this beautiful picture it is not easy. And, why do you want to get only this that is your next question right why am I interested in this? You are interested in this, because you want to ensure let us say you are the car, you want to ensure that you are always moving in a straight line right.

Unless you know where your left edge is and where you are right edge is, you are not going to remain straight right. Remember the other algorithm that we showed a simple algorithm, where there is a sensor and there is an actuator. Let us assume that in this case, in this particular example camera is our sensor and actuator is indeed the steering wheel. If, you want to remain straight, your camera sensor should keep telling you where your left edge is, where is your right edge is, in which lane you are and in what direction you will proceed?

Suppose in the road curves like this right, then you will come here, and then this sensor will give a command to the steering sensor. That it should turn also to the left. So, that it continues to remain in a straight line as much as possible right that is only way. So, your goal should be to ensure that you are always in a straight line, which means this edge and this edge you have to detect accurately. So, that is what our whole exercise in this courses, how to get to this edge as accurately as possible and how to detect these lines. I have been using these words.

So, if you want to get these lines you have to do 2 things; a you have to do edge detection and after you do edge detection, you have to do line detection. You have to do edge detection and then line detection; a has to be done first then you follow it up with b, a leading to b edge detection leading to line detection, then you will get this line perfectly in order. Now I have motivated you well enough. Here, the trouble is if you look at the section edge detection. This one is a rich area of research by itself it is a rich area, extremely rich area and there have been developments over the last 50 years of development.

Therefore, it is hard, this edge detection is an extremely hard problem and people are all the time after this. Because, if you can do this edge detection accurately and you can do line detection, then many problems in autonomous driving are actually solved. Therefore, several algorithms are available for edge detection. Out of so many of these algorithms you may want to choose one candidate algorithm, which is of interest to you; candidate algorithm for edge detection. Now, you may ask how do you choose, what is a good edge detection algorithm? Obviously, when you say it is a good edge detector it should do good edge detection obviously, what does it mean? It should minimize false positives and false negatives very important.

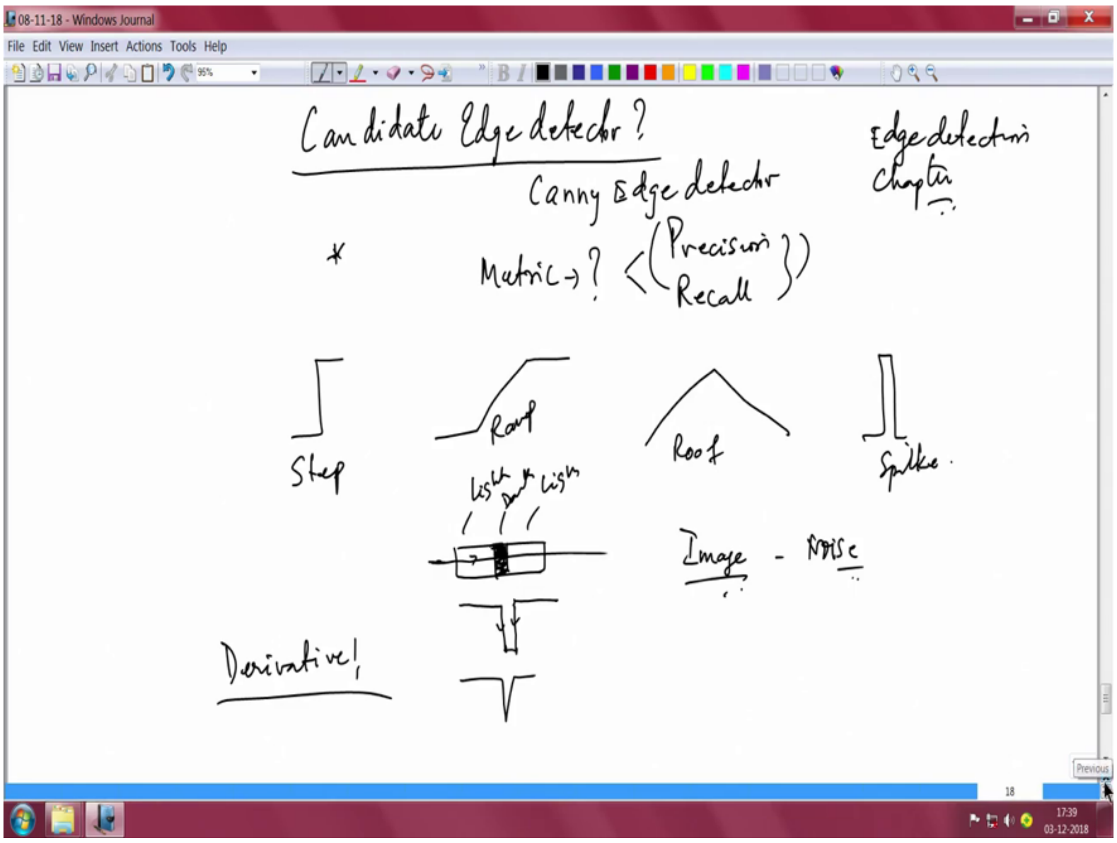
So, if you say candidate ; one of the requirements (a) is minimize false positives and false negatives, what is b? B is good localization, what does it mean? It simply means that detected edge should be shown exactly to its location; other than shifting in any direction, if it gives me an edge shifted left or right you can now imagine how disasters it can be?

This is actually beyond this good road edge, you do not know what is there it could be a ditch, it could be simple mud untarred part of the road there can be trees right, there can be trees all this or it can be anything. So, unless you have extremely good localization, you are going to have a problems.

So, it should have good localization and it should just indicate one single point just one edge it should indicate, it should not indicate for every point here or there , should not tell me that there are multiple edges, this is not acceptable, it should always say that there is only one edge for a single point, this is also another important thing.

So, this is what you should be looking at.

(Refer Slide Time: 14:48)

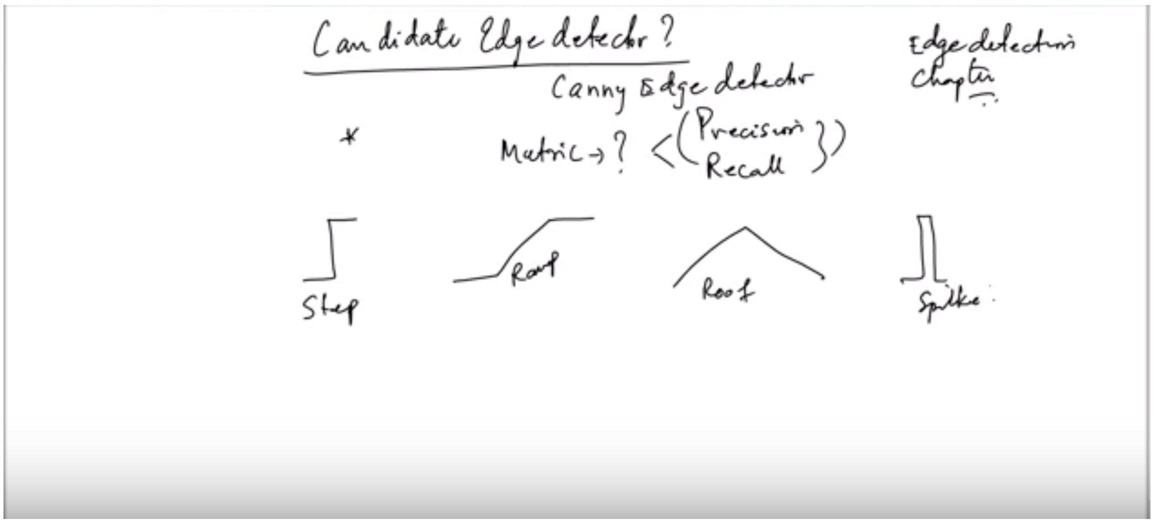


Now, the question is what is this candidate algorithm? What is this candidate edge detector? Well there are several books and several papers and several algorithms on this one topic, I found a good book which I am sure you will be able to download on computer vision. So, I will write, it is called fundamentals of computer vision, by Professor Mubarak Shah. I think it is called fundamentals of computer vision, please have a look if you Google for this you will find this book. And, I would strongly recommend you to read the chapter on edge detection.

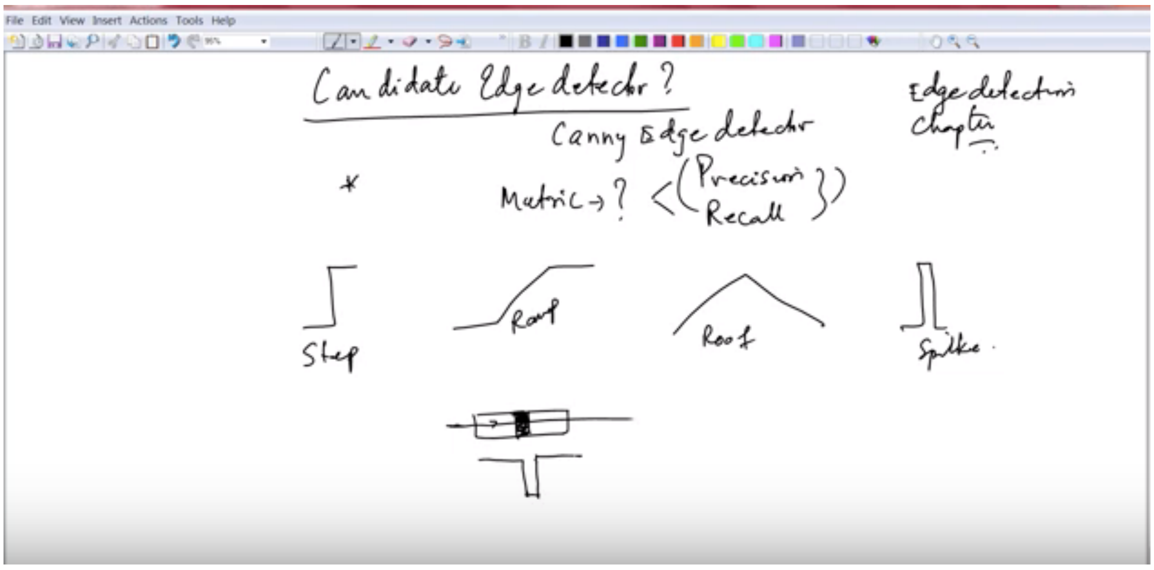
This is not a full course on edge detection, I am just trying to motivate you towards our IOT examples. Therefore, you must spend sufficient amount of time understanding the candidate edge detector. The candidate edge detector most popularly used today is canny edge detector. So, if you look at what the main purpose behind an edge detection is, if there are any abrupt changes it is essentially an edge isn't it? Any abrupt change is indeed a edge, any sudden change in image. Image will be of one type till certain point and suddenly you have a change and therefore, there it is indeed a candidate edge.

So, any sudden change or discontinuation in an image or discontinuity in an image is essentially you can declare it as an edge. It is a very simple definition, but then if you read that book by Professor Mubarak Shah and he himself will explain, that when you talk about edges you talk about surface normal discontinuity, depth discontinuity, surface colour discontinuity, and illumination discontinuity all of this can potentially create edges. So h do you measure which is a good edge detector, how do you know is there a metric to say that this is a good edge detector or this is not so good and all that. So, there are 2 terms associated with edge detector; one is called precision and the other is called recall.

All edge detectors are essentially measured using these 2, what is the precision of edge detector? And, what is the recall of the edge detector? And, you can have a step kind of edge detector, you can have ramp kind of edge, you can have a roof kind of edge, you can have a spike kind of edge. So, all these type of edges can exist in an image. So, again I have picked most of it from Mubarak Shahs book. So, if you read that book or look at his notes on the web you will find all of this. So, it is extremely useful and quite intuitive for you to understand.



Candidate Edge detector ?
Canny Edge detector
Edge detection Chapter
*
Metric → ? < (Precision Recall))
Step
Roof
Roof
Spike



So, essentially he shows a nice picture and he explains that the complete thing is darkened shown in the bottom part of the picture. You can draw a pulse line. What does it mean? It means it is all bright till line is straight, if you go from left. Image has essentially 2 dimensional right. You go x and you can go y. So, if you go from left direction it is all bright, suddenly it goes black it goes dark, then again it rises back right.

So, essentially you will get a signal like a rectangular pulse. Now, one simple mathematical way to detect that there is an edge is to differentiate this, if you take the first derivative you will get something like inverter triangle in line.

So, start using this derivative both in the x direction as well as y direction, and then detect edges nicely. However, any image if you take, you take any image it is not easy to process the image directly. First of all there will be lot of noise in the image. So, you have to remove the noise and typically what they do is so, you essentially make the line a little thick, but you may now ask if you make it thick would not it look blur. Yes, it will look blur this will lead to a blur that is right, let it be a blur, but at least you have smoothened it. And, after blurring you still have a beautiful edge which you can detect.

Therefore, what people normally do is particularly if you look at canny I am not going into details of all types of edge detectors, you apply the you before you do any first derivative or anything before you apply the derivative.

(Refer Slide Time: 23:05)

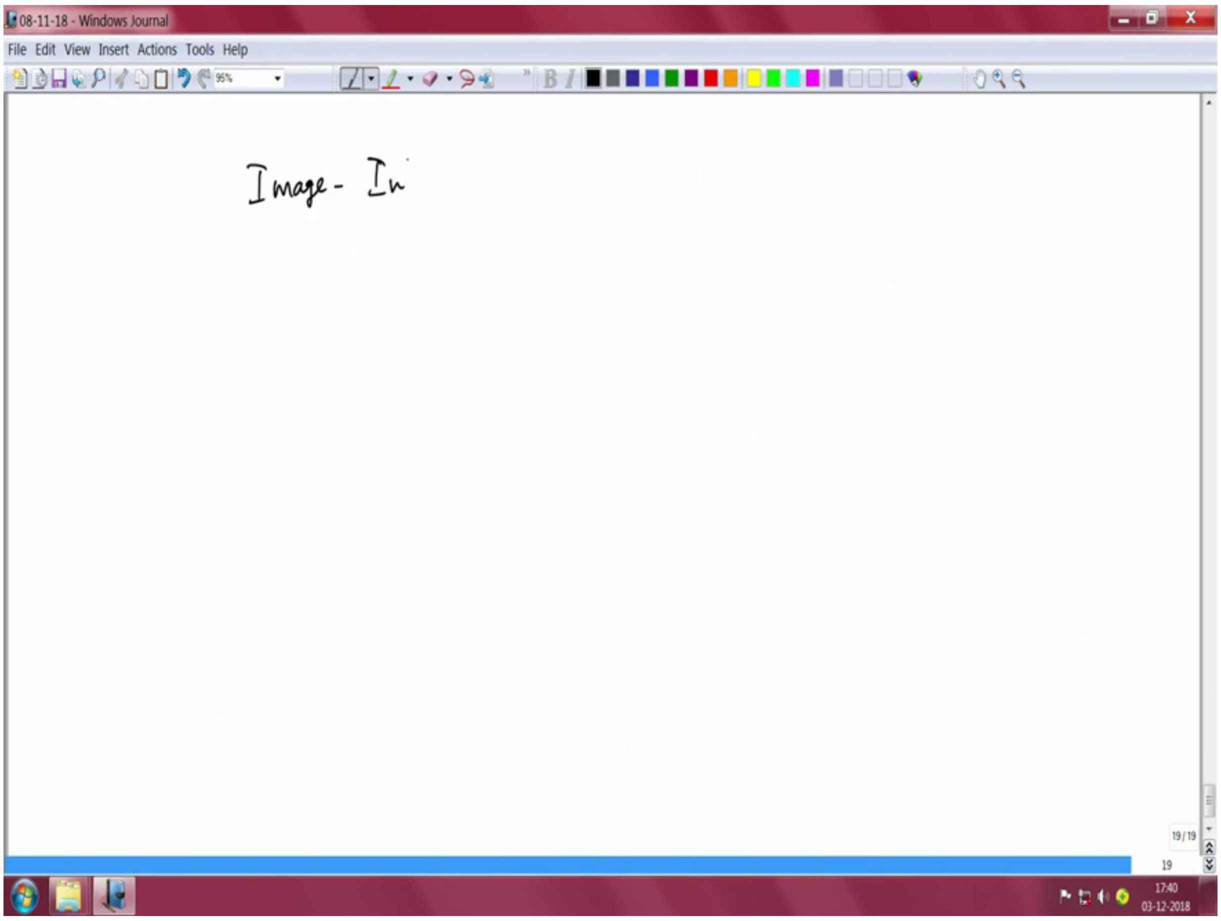
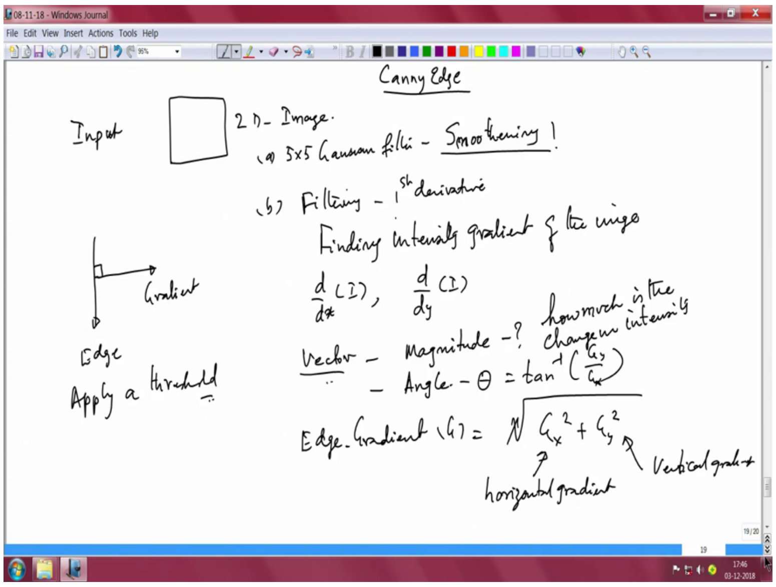


Image - In

You essentially, you take the image, image is your input right, I will input image 2 dimensional please note x and y 2 D image.

(Refer Slide Time: 23:15)



Canny Edge

Input 2 D - Image.

(a) 5x5 Gaussian filter - Smoothening !

(b) Filtering - 1st derivative

Finding intensity gradient of the image

$\frac{d}{dx}(I)$, $\frac{d}{dy}(I)$

Vector - Magnitude - ? how much is the change in intensity

- Angle - $\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right)$

Edge Gradient (G) = $\sqrt{G_x^2 + G_y^2}$

horizontal gradient

Vertical gradient

Gradient

Edge

Apply a threshold

And, you apply a Gaussian filter typically they choose a 5 * 5 Gaussian filter, very simple you take neighbourhood pixels of 5 cross 5 and substitute every pixel with the average, that is how you get this? you will get an average number replace each one of this pixels with that average and then go on doing it you will get the blurred image.

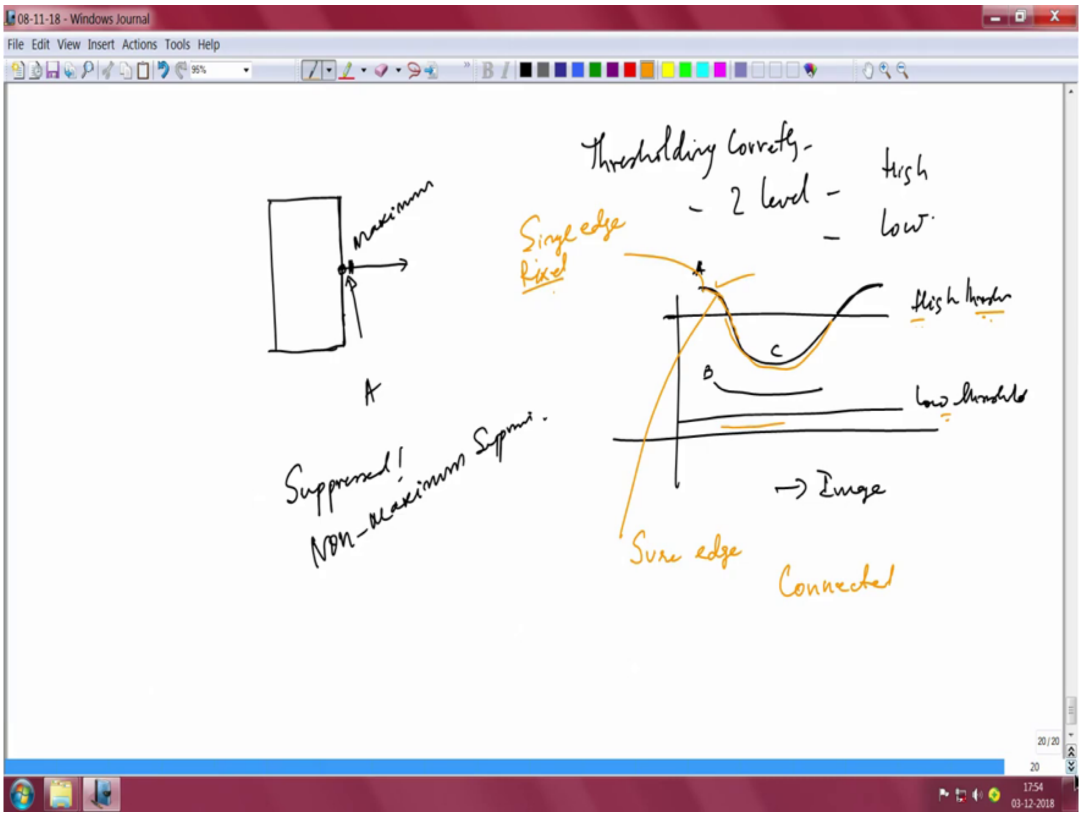
And therefore, you get a smoothened so, when you say Gaussian filter this is mainly for smoothening. So, remember first step is smoothening. This is input right, this is input and here is your step a. Let us come to step b; step b is applying what is known as a filter. I am just talking of canny please note I am only talking of canny edge, by the way this smoothening is common to all type of filter. The canny edge is special because it does several interesting things so, let us see smoothening is done then the next step is what you known as a filtering.

Essentially when you say filtering what you are interested in is, finding the intensity gradient, finding the intensity gradient of the image right. This filtering also includes getting to the first derivative that is important, filtering plus first derivative, you have to do both. Then essentially when you are saying first derivative, you are saying d by d x of the image I. d by d y of the image which essentially leads you to generating a vector. Essentially you are generating a vector, it has magnitude components and the angle component, essentially you have the magnitude and you have the angle. So, angle is theta magnitude essentially you are saying how much is the change in intensity.

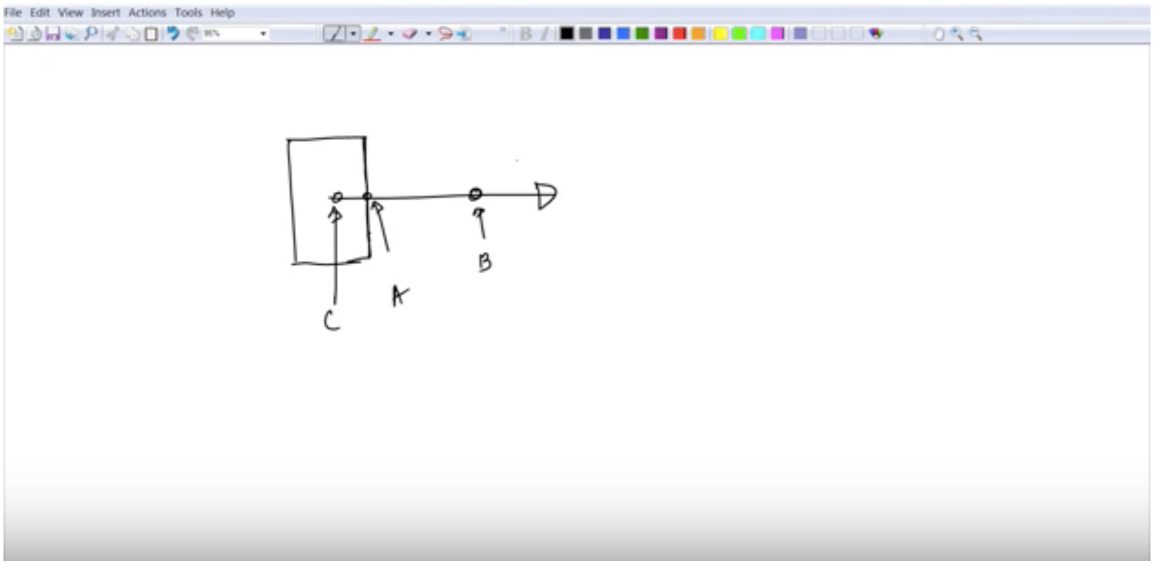
You have to quantify this right. So, typically edge gradients essentially you are trying to get to the intensity gradient, and this edge gradient G is computed in a very simple fashion Gx square plus Gy square. You know what G x and G y are this is the horizontal gradient, and this is the vertical gradient. Since we said gradient, and it is a vector that we get the theta is quite simple this is nothing, but tan inverse of G y by G x.

So, now this is all you have to do in canny edge all of this is done by one single function in OpenCV. However, you must know it properly right. So, that is a reason I am trying to explain this. Anytime you have an edge the gradient is always perpendicular to it gradient is always.

(Refer Slide Time: 28:56)



Now, on the gradient you apply a threshold. I will show you with an example, again you could find them in several books, I will show you a very simple example by I will sketch it and show you so, that if you take the system.



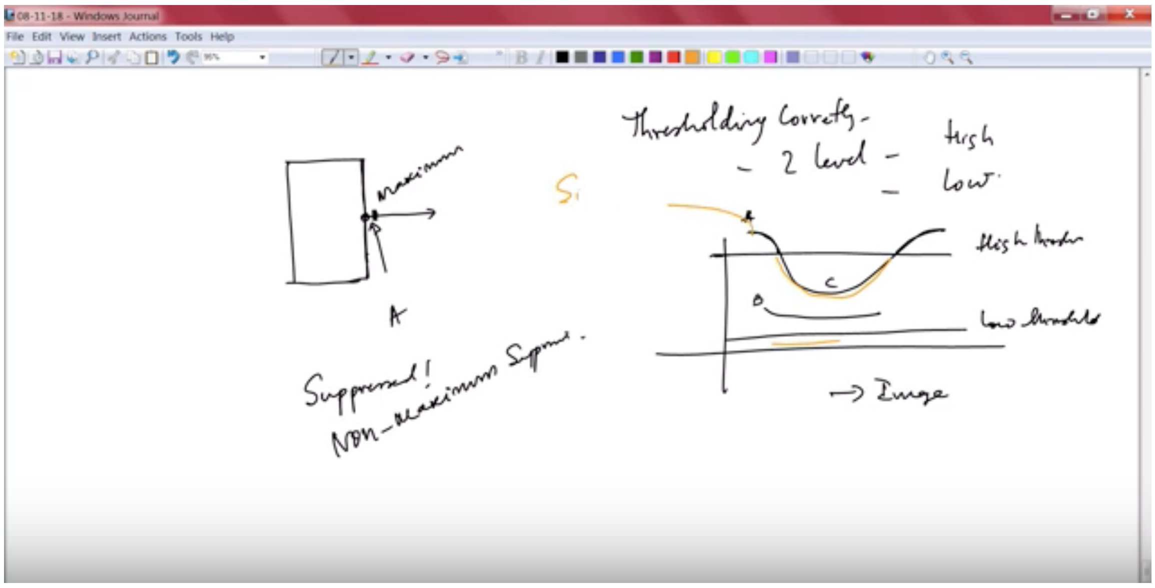
Now there are 4 edges. Consider this edge. The gradient is always perpendicular to the edge we already said this. There are 4 pixel points, let us call this C, A, B, you know

now that it is very important that you have 3 points and you do not know where the edges, why because this is the gradient direction. In the gradient direction there are 3 points and you really do not know which essentially is the pixel that you have to worry about, which means C and B have to be suppressed. So, the next step in canny edge is suppression of non-maximum.

So, then if you suppress C, B you are only left with you are left with essentially A and that indeed is the point of interest, pixel of interest, edge of interest, and you have to now say; this is the point I wanted to pick and you do this and then you find out essentially, that the edge has been found at this point.

So you have to do this all along the complete image and then you have to essentially ensure that you have detected the edge correctly. This is not over, problem is not solved, because you may get another very close point which is also around the same gradient, far away points no problem what about nearby? that can create another mess for you, which means you have to apply thresholding correctly, which means you normally do what is known as 2 level thresholding which is called high and low thresholding.

Unless you apply the 2 level thresholding you will not be able to get rid of those which are extremely close to the edge pixel at all. For that you can look at again several examples exist, I picked one from explanation related to the canny edge. Here this is the high and low threshold. Now, look at this picture.



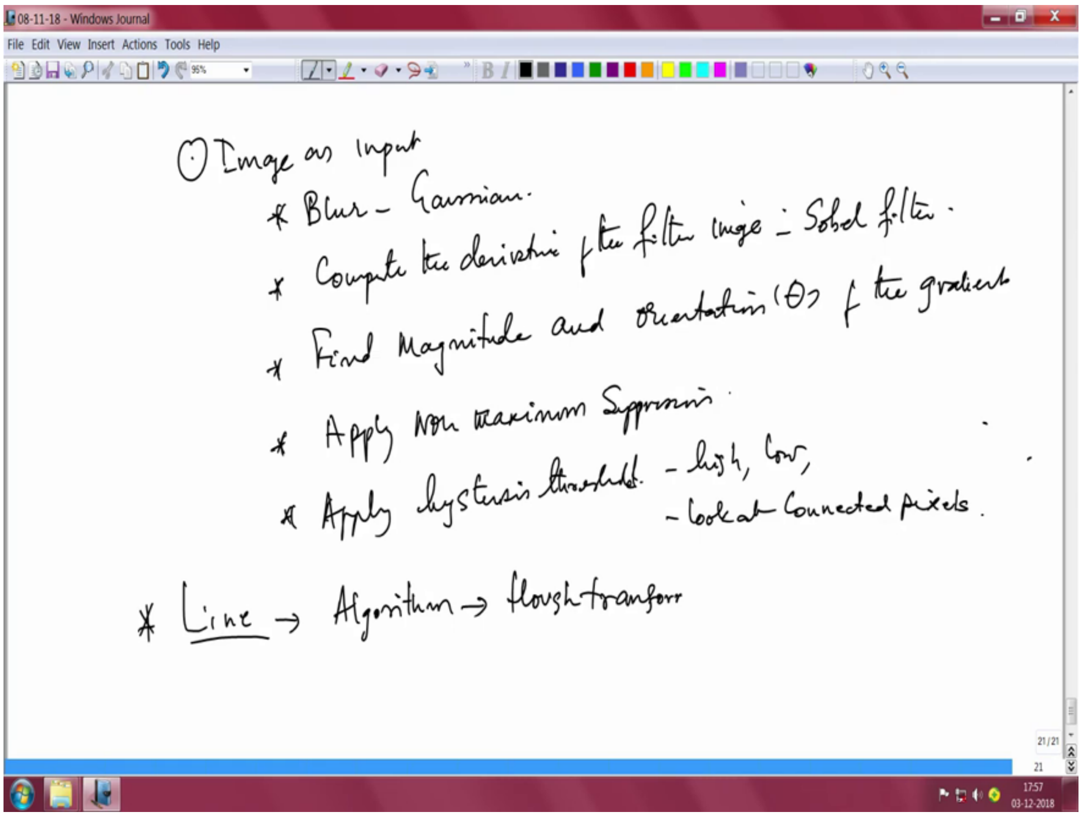
The idea is pretty straight forward if you apply a single threshold, suppose you apply the high threshold ,What will you get as an edge? you will only get starting and end points of Edge A, which is not true you have lost C part.

Now, how to differentiate between this system and this? take a single edge pixel at A and then start looking at it start following it.

If, it continues to be part of the line then keep following it, but remember you will have to start with the single pixel which is above the high threshold. If there is a pixel above the high threshold keep following it and then trace it through. So, automatically C will come through A edge will come through, but B will not because it is below the high threshold. In this manner one could actually detect the fact that A as they call is essentially a sure edge, A is actually a sure edge right? because it is above the high threshold and C below the high threshold, but as your following or nothing, but as it is connected you can say it is connected.

You are following meaning you are connected you are connected, because C is connected, you essentially declare it part of the edge and anything else is simply discarded. So, the question will be how do you choose the high threshold and low threshold, and that is something that you could really experiment with when you start coding for using a canny edge detection system. So, let me summarise the whole process of canny edge so, that you will be able to experiment with canny edge successfully in any code that is given to you right.

(Refer Slide Time: 36:48)


08-11-18 - Windows Journal
File Edit View Insert Actions Tools Help

⊙ Image as input

* Blur – Gaussian.
* Compute the derivative of the filter image : Sobel filter.
* Find Magnitude and orientation (θ) of the gradient
* Apply Non maximum Suppression.
* Apply hysteresis threshold. – high, low,
  – look at Connected pixels.

* Line → Algorithm → Hough transform

First step is you take the image as input then what you do? apply Blur, I will simply say Gaussian blur, apply a Gaussian blur. Once you do this then you would find the compute the derivative. Compute the derivative of the filtered image, this is typically a sobel filter in canny edge. Then find magnitude and orientation theta, theta you have to find of the gradient.

Then, what you do apply non maximum suppression, then apply hysteresis threshold. By looking at high, low, look at connected pixels. This completes the whole operation of canny edge. Next important step is trying to find the line and for this we will have to look at another important algorithm and that algorithm indeed is the Hough transform. So, one way of you know sort of blurring this edge is to do an averaging as I mentioned, but in canny edge what they do is they take a Gaussian blur. So, this blur actually is Gaussian blur, and so, you take the original image do a Gaussian blur apply the sobel filter, then do non maximum suppression and then follow it up with the hysteresis thresholding, 2 level hysteresis thresholding.

So, let me quickly come back Gaussian blur, then complete the derivative of the filter and apply the sobel filter with sobel filter find magnitude and orientation theta of the gradient apply the non-maximum suppression, apply hysteresis threshold, and then you do Hough line transform.