Modern Computer Vision

Prof. A.N. Rajagopalan

Department of Electrical Engineering

IIT Madras

Lecture-51

We are at a descriptor sift, we are talking about how to arrive at a descriptor of sift. I think I just started talking about it during the last class. So, I said that this is a 128 dimensional vector and it involves a bunch of steps. It is a first of all so around the interest point at which we have already seen how to arrive at the interest point and there is a definite, you know, dominant orientation. So around interest point a 16 cross 16 grid is chosen. Now this grid, okay, that does not mean it is, you know, so the grid size itself whether it involves 16 cross 16 number of the pixels or whether it involves more number of pixels depends on the scale.

Okay, now this is somewhat like a, like a kind of, you know, hyper parameter. Okay. So it is not very clear, I mean nowhere do they tell us to what should be the scale. I mean, right, so the only thing that we know is if you go up the scale then this area will also increase.

Okay. So maybe you can just, you can just kind of make it a function of σ. Right. The simplest thing that you can do is multiply it with actually, with the kind of value of σ that you have. Because I have tried, you know, looking for this but that specific information that even in the octave or the open CV implementation nobody tells you exactly what is that, what is that, even the original paper does not say, it just says, you know, keep it as a function of the scale.

Okay. I think, you know, it does not probably, it is not very sensitive to that. A 16 cross 16 grid is chosen. So grid, okay, this is not like pixels. And the gradient norm and orientation, this you know, the gradient norm and orientation are found for each of the pixels, within that grid found for each of the pixels.

Okay. Then there is a Gaussian weighting that is done, okay, on this norm because something that is far away, because this grid can be really, can also be big, right, because it kind of, you know, changes with the scale. By which I mean that the extreme point at whatever scale you found, okay. So by the 16 cross 16 grid we mean that at that scale, right, at that scale it could be a large region or found for each of the pixels. And okay, I think I probably already said last time to achieve rotation invariance, okay for rotation

invariance, you kind of recompute the angles with respect to the dominant orientation, with respect to, which is the orientation of the, you know, key point, with respect to the dominant orientation of the key point.

Then you do some kind of Gaussian weighting now. So the norm of the gradient is actually Gaussian weighted now, the norm of the gradient. So what this means is that depending upon how far away you are from the key point, you sort of reduce the, reduce the, what do you call the, no this one, the weight or the value of the norm, the norm of the gradient is, or the strength of the norm, of the gradient of the norm, norm of the gradient. To achieve rotate their angles, the norm of the gradient is Gaussian weighted, Gaussian weighted, again right I think there is, this one is something like 1.5 σ is what they say, okay, this is again a hyper parameter.

So if you are at that scale right, then you take a Gaussian weighting of 1.5 σ. Again I am not going to write it down because these are all I think in a different implementation claim, different thing, somebody will say 2.5 σ, they say Gaussian weighted by the, and again okay this weighting depends on the scale, depending on the scale, well not exactly right because it is like 1.5 σ right, so it is not, so let me write this as, I will just write approximately 1.

5 σ, this one, this grid is dependent on the scale. This is dependent in the sense that it is already like 1.5 σ right, so this one is what is dependent on the scale. This Gaussian weighted by the distance of pixel to the shift point. See that is also the reason, this is also the reason why for example right, you know generally if you implement a vision code right, seldom will you find that you can take something off the shelf and use it.

I do not know how many of you have tried it out, this is also the reason why you know, why it took a long time in order to make these things really robust and all. It is all these you know, even though the theory may look good but in a practical implementation right, when you actually use it, when you cannot directly use, you know directly take it from a MATLAB code and start using whatever images you have, you always invariably have to tweak something. Yeah, this is always been an issue, this is unlike your you know a communication or something right where you know things are kind of well laid out right. So that is always been an issue but now with these deep networks and all the robustness is gone up like you know multi fold but this traditional methods right, there is always been you know somewhat, somewhat of a you know somewhat of an issue you know, a ticklish issue, it is not something very serious but yes, but these things matter because if you think that you know I can just take an image of this room and then I take another view point, I can automatically match, I can run you know shift. Now very likely that simple is yeah, simple problems that you can solve but in the moment right the problems become more

complex you know it is normally not true that you can take something off the shelf and use them.

So you will invariably have to tweak something. The 16 is then divided, the 16 cross 16 grid is divided into 4 by 4 cross 4 blocks into 4 cross 4 non-overlapping blocks, non-overlapping blocks. So it is like saying that you have a 16 cross 16 grid, so you do it like 4 cross 4 right, non-overlapping blocks divide them and then an 8 bin histogram, by the way this orientation based histogram that we are creating right, this is itself is actually you know in the next class when I talk about HOG okay that is actually based on this kind of an idea only. So then 8 bin orientation histogram, orientation histogram quantized at 45 degrees again that is what I am saying right, so these are all things that you may have to tweak okay but of course the fundamental sort of a descriptor is this, is computed for each 4 by 4 block okay. So then what this actually means is that you have already weighted the norm and all right, so for example if you are at the center then something far away from the center it is this one, the gradient strength has already been sort of weighted down and now we are right within kind of each of these blocks now we are actually creating these orientation bins which is the same array that we said.

So for example you know so 45 degrees means you will have, we will have kind of say how many bins you will have, 8 bins right, so you have an 8 bin histogram and then right now you start to sort of see right which, so wherever if some sort of you know a pixel has a you know for a certain orientation it has a certain gradient norm, you actually put that into that bin and then you add up right, so that you get again you know sort of you know 8 bin orientation histogram for each and there are of course you know 16 such blocks right, it is computed for each block, for each 4 cross 4 block, for each block. So and since there are 16 such blocks and this ordering at all is important okay, the way you order for example see what you do is you do simply you know this one a concatenation, so for example you have an 8 bin orientation histogram right, let us say for this you have an 8 bin, for this right you have an 8 bin histogram right, then when you actually concatenate them you got 16 of these right, so you know concatenate them into one, that is what I mean by 128 dimensional vector right, so you have to concatenate them. So when you concatenate you concatenate the same way right every time okay, that is how you cannot just change the order, so since where is that, so this we get okay since there are 16 blocks you get you know 128 dimensional feature vector or a descriptor in this case one dimensional shift this one a descriptor. So what this means is that right this is able to extract information in and around that feature point that actually explains that region okay, explains that region in a manner that for example when you match right, so the whole idea is that if you directly try to match intensities right it would not simply work okay right, that is the whole idea behind going after features and all right, if you have just 2 images you cannot simply take the image intensities and start matching right, so features are far more robust because you are

kind of extracting a lot of things that you believe makes for you know gives room for invariance for a lot of things. It is a 128 dimensional descriptor which is the okay, now and then of course this is further normalized to you know to actually unit length, this further normalized right, prior to normalization clearly I mean you know that there is rotational invariance because of this right, because of the rotation where you kind of recompute the angles right with respect to the dominant orientation.

Therefore, rotational invariance you can see is already inbuilt to the you know into a descriptor, the other thing is scale invariance we know is already coming because of the way you actually solve for shift itself right, solve for the features and therefore right and therefore I mean so you can see that you know the fundamental things that you want out of this in terms of the invariance for the geometric invariance right, it already satisfies things that you would want it to satisfy. In addition for the photometric part right this is further normalized, of course translation variance is obvious right. So this is further normalized for photometric variations or intensity variations okay, this further normalized to unit length for photometric variation, so it is like saying that you know if you had a contrast increase right for example if you had like you know A times I, of course you know if you had an addition that is that anywhere it takes care of because you are doing a Laplacian which is like dog which is like a derivative, so any addition is okay you can handle no problem but if it is a contrast increase right let us say you have some like you know multiplication of A right then the idea is that right I mean because you have gradient will also get scaled by that factor if you simply normalize it outright then it should not matter when you actually match. But this is only if there is the illumination sort of uniformly going up right, for non-uniform illumination which you can have for example non-uniform by which you mean that suppose I take this image right so whatever is falling on this side is very bright right and then that side is not so bright, so you have a local sort of illumination which is very high, so in such cases right this cannot completely handle it but then what they do is you know they normally they sort of you know put a cap on how much you know for example after you are free you normalize right so you have this 128 dimensional vector right, so which you have kind of say right you know which you have done a normalization, now which means that if you sum up all these values right it will be equal to 1. So what they do is you know the maximum that they allow is actually 0.

2, this is again an implementation thing, so the value right cannot kind of say exceed 0.2, if something exceeds 0.2 then it is frozen to 0.2 you do not allow it to go beyond that simply because it could be happening because of some you see non-illumination variations and then of course you will have to say renormalize that you know how to do right I mean it is like saying that you freeze everything and then you add up all the quantities right and then you simply scale every number by that sum, so that the sum again sums to 1 but now you have you have sort of capped the capped you know you can just say contrast

sensitivity. It is all this is more like empirical okay, this is not like this is not a strict you know strict theoretical result that if you do this then you can because non-illumination variation itself can be of different types you know there could be specularity and also it is not a claim any of that, these are all things that they found out right you know works in practice and the you know interesting thing is even though even though right strictly speaking this is a you know sort of this is not even meant to handle affine transformations right but in reality right people have found that this works extremely well I mean under many many conditions okay in fact you should do this just take right go outside wherever you know hostel or wherever inside the campus wherever you are take your cell phone take let us say right 2 images of a scene today you take 1 tomorrow again when you go back I take another image do not take from the same view point try to see right I mean how many matches you get and take any shift you know off the shelf code and run it.

Of course you know see that there were further you know in further improvements over the basic sifter say even something called an affine sifter and all that is basically tailored to handle you know affine variations and all we do not we do not actually go into that in this course but the point is even the basic sifter you will be surprised even you should even try something like you know day and evening you know even try that right take something that is like bright you know in the morning and then come back in the evening 6 o'clock take do not take a very dark or something but maybe right you know this one a dusk time right you take and then you should again run and see right you will be actually surprised that it works so well and it is very fast the implementation and all right and the and and you know that slide that does not have very good examples but I think you should try it yourself why should you have to accept right what I show here right why do not because these are things that you can try yourself right it is very easy to you are not implementing it anyway you are just taking an off the shelf code running it on images just to see whether the comparisons are good whether the matching is good sometimes you would not be able to match but then this would have matched it well. I think I showed a Mars image remember but it was very difficult for us to tell that there is a match okay so much for SIFT okay let us go to the next one which is called SURF right this was that was 2004 right so this is I think 2006 this is actually came as an came as an improvement to SIFT this came as an improvement over SIFT basically many of the ideas are borrowed from there the authors are not it is actually a different different these are different authors so this guy is some Herbert Bay and Laquan Goul is so this group is very good okay this is headed by this chap so 2006 okay. So what does SURF stand for it stands for it is speeded up robust features okay and the main idea is to speed up things okay I mean even if it comes at some at some cost of at the cost of making some approximations okay and then it actually runs very very very similar to SIFT but then but the idea right that they have used is actually very nice and it is very interesting right how they actually how they solve this problem so we will see okay so improvement so basically you know it is as an efficient

alternative to SIFT. In fact right this implementation also is available okay if you can even try this in addition to SIFT but of course you know you would not notice the speed difference if you if you should not alternative to SIFT so the main thing is right instead of trying to rely on a Gaussian function I mean I told you right I mean I think in one of the classes I told you why let us say a Gaussian has been the ideal pick for let us say several reasons one is of course on all the math that showed you know how we can pick you know from a log the extreme and all that but in addition to that I also said right I would say scale up the lower features will start to go away only the coarser features remain no artifacts get introduced actually I mean if you really read about it right the actual theory says that for a lot of 1D right this has been proven but for 2D it does not look like right there is so much of you know sort of theoretical support people just believe that whatever is happening in 1D should also most likely happen in 2D. So the argument that these authors make is that you know because of the fact that right it is not yet completely proven that you know that a Gaussian for a 2D case is still the best so go ahead and approximate they say okay fine right let me do something else and the process right it is not like a bad approximation but it is a very smart approximation and they actually show that right if you were to solve it using what are called box filters.

So box filters the idea is to use box filters as approximations to the Gaussian derivatives right. Now you may wonder so box filter is typically like what I mean on all 1s so something like this I mean typically read by box you mean that it is a kind of you know a contiguous a contiguous array of numbers right so you can have something like what this is like 4 cross 4 and so you may have like 1 by 16 all 1s right this is actually a box filter and you might wonder right I mean you know why such a thing makes even why does it even make sense right that you have something like this you know which seems like a big deviation from a Gaussian. But this actually goes back even more interestingly this goes back to an earlier work in 2001 I think there is another paper I do not know how many of you have heard about the Viola Jones phase detector right so this is one guy so there is something called a Viola Jones phase detector okay now this now this they so they had a paper in 2001 where they actually talked about something called an integral image. So these actually so these people actually know so this so this group borrowed that idea and they actually showed you know and then they started from there in order to show various instincts.