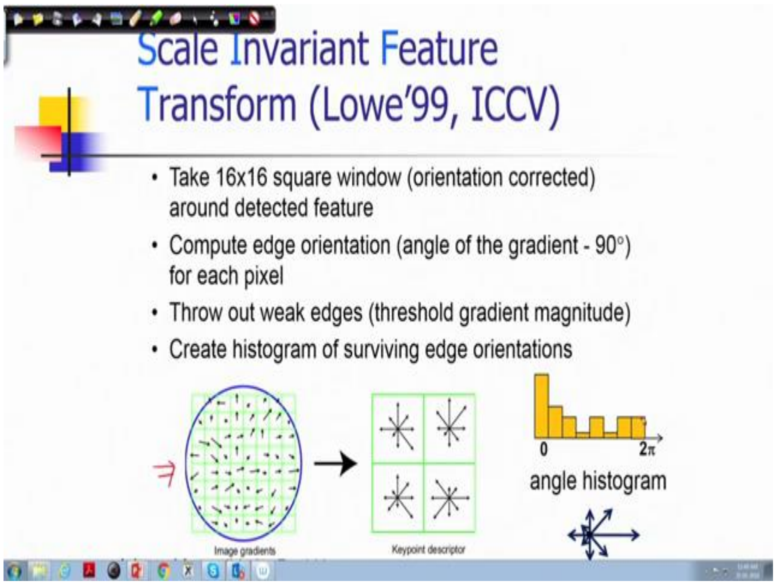
Computer Vision

Prof. Jayanta Mukhopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 26
Feature Detection and Description Part – III

We are discussing about the detection of features, feature points and also how do you describe those feature points uniquely, so that both Detection and Description becomes transformation invariant.

(Refer Slide Time: 00:33)



Scale Invariant Feature Transform (Lowe'99, ICCV)

- Take 16x16 square window (orientation corrected) around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations

In the last lecture, we discussed how a feature point can be very distinctive robust to the transformation of scale and rotation, translation, and they are characterized by their position scale and also orientation.

Now, we will be considering a descriptor, which is a very popular descriptor used widely in the image processing techniques and this descriptor is called scale invariant feature transform. It was proposed by David Lowe in 1999. In fact, these slides are adapted from his presentation and we will be considering how these descriptors are not, right.

So, we can consider once we detected a key point. So, around that key point we can take a 16x16 square window. You should note that it is orientation corrected; that means, once you have the orientation dominant orientations around a key point then you perform the rotation to align it with your reference axis, for example, x-axis. And rather the x axis to rotate it along this direction. And then in that rotated image now you take a 16x16 square window. And let us see how these descriptors or local statistics is accumulated. In fact, there are some diagrams I will be explaining them.

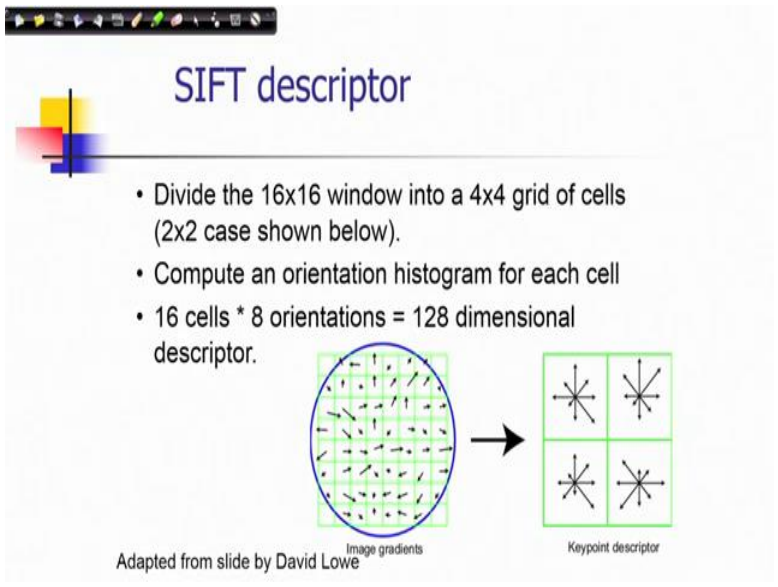
So, what we need to do? We need to compute edge orientation; that means, angle the gradient minus 90 degree. So, that is how the edge orientation is defined. So, the gradient operations in the very first lecture itself I discussed that, you have to compute the derivatives along x direction and derivatives along y direction. So, these two vectors would give a direction and that is angle of the gradient. And then if I subtract 90 degree from it you will get the edge orientation for each pixel.

And then you can throw out weak edges; that means, the magnitude give a threshold over the magnitude. And then create histogram of surviving edge orientations which are shown here figuratively in those diagrams. You can see here that typically in this picture it is showing that in that 16x16 window at every pixels how it is shown of course, in a smaller window.

How the directions are shown at every pixels; that means, at every pixel you computed gradients and these are the directions. Some of them are thrown out from the consideration of the magnitudes if they are weak. And then you bring those directions into a histogram which I discussed earlier that because directions can be considered a range it ranges from 0 to 2π radian angle, and then you discretize those intervals. Each interval is called a bin. So, in this case as you can see from the diagram itself, there are 8 directions.

So, there would be 8 such intervals each by 45 degree (π/4) and within that you are binning those directions and then now you can get the histograms. So, the length of the vector in this particular diagram is showing the magnitude, that is showing the number of pixels or even you can add those magnitudes of a gradient to get the magnitude of this histogram to get that angle histogram.

(Refer Slide Time: 04:34)



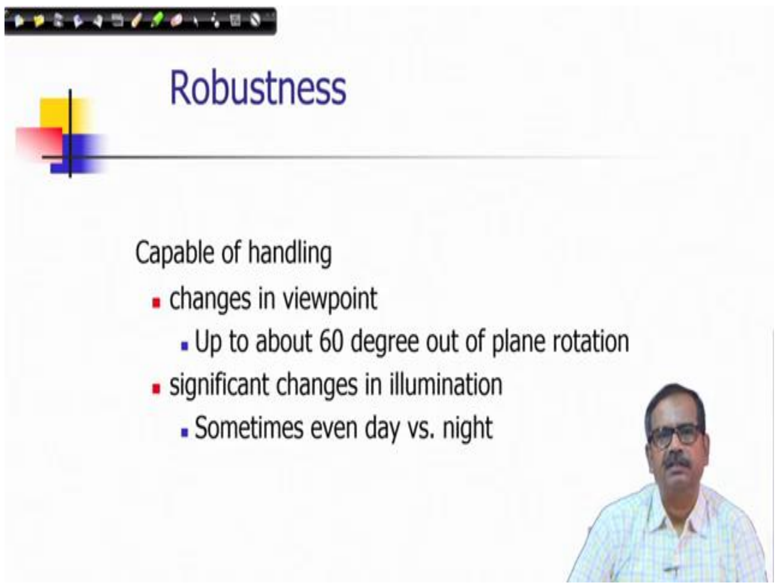
SIFT descriptor

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below).
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor.

Adapted from slide by David Lowe

So, next what we do that now you can divide this 16x16 window into a 4x 4 grid of cells. So, in this case it has been shown 2x2 in this particular diagram for typical examples and for each grid of cells you get the histogram. So, compute an orientation histogram. So, what I discussed is the histogram of composition of the orientation histogram that was not over 16x16 cell. It was each of 4x4 grid of cells and each one will have 8 bins of histogram. So, each one will give you 8 dimensional vectors corresponding to each histogram. So, since there are 16 cells, so finally, if you concatenate all those vectors into some order that would give you 128 dimensional descriptor. That is what is your sift descriptor.

(Refer Slide Time: 05:26)



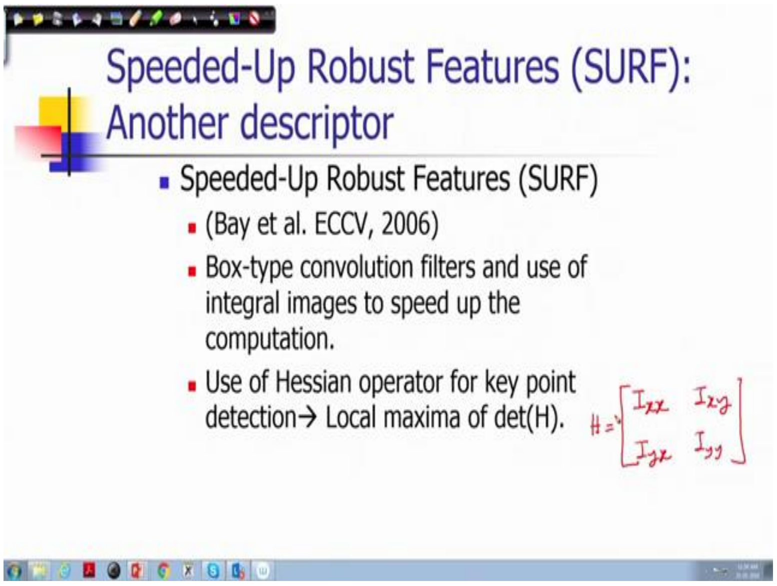
Robustness

Capable of handling

- changes in viewpoint
  - Up to about 60 degree out of plane rotation
- significant changes in illumination
  - Sometimes even day vs. night

So, the properties of sift descriptor is that it is capable of handling changes in viewpoint and there is a significant changes in illumination, just to qualify that it even it is observed that it can handle up to about 60 degree out of plane rotation and it is found to be invariant even the illumination variations due to day and night.

(Refer Slide Time: 05:55)



Speeded-Up Robust Features (SURF): Another descriptor

- Speeded-Up Robust Features (SURF)
  - (Bay et al. ECCV, 2006)
  - Box-type convolution filters and use of integral images to speed up the computation.
  - Use of Hessian operator for key point detection→ Local maxima of det(H).

$$H = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix}$$

So, this is what about this sift descriptor. But, now there are various other descriptors. I will briefly summarize them in this particular lecture. One of them followed by sift descriptor is a speeded up robust features or SURF descriptor which has been proposed in 2006. And this descriptor also is very popular because there are various modifications which have been introduced in computing this descriptor, so that it could be computed efficiently. That means, your compressional speed increases significantly using this descriptor.

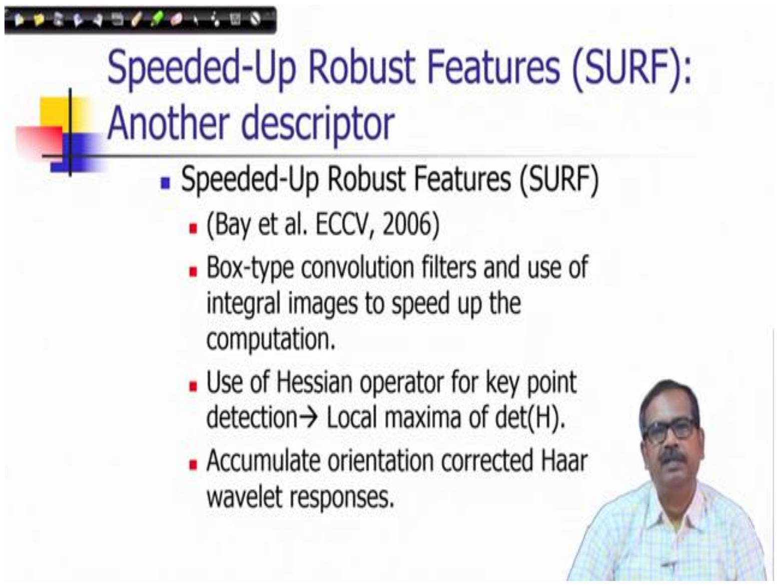
Mostly, in the sift descriptor the major computation that we need to carry out by using the Gaussian convolution. So, in this case in surf descriptor, instead of Gaussian convolutions what we can do? We can perform a box type convolutions; that means, we can compute the same gradients or and also same double derivative. So, we will show how. What are the detectors that have been used in this particular case and they convert using a box type filters. I will explain what is meant by this box type of filters.

And in fact, which means in this case the convolutional masks they have only weights of say 1 and minus 1, and there are patches where all are 1 and all are minus 1. Then those patches are square patch and that is how this kind of the shape of this filter is called box type of filters. And these computations can be carried out using an integral image which again I will explain in this particular discussion.

So, the key point detection using this descriptor is performed using Hessian operator. Instead of the difference of Gaussian operator we consider the Hessian operator of the image itself. So, Hessian operators are those no double derivative operations in the direction. So, in this case for an intensity images, you perform double derivatives along x direction then along x and y, y and x, and y directions. So, this is what is Hessian operator. And you perform the local maxima of the determinant of this Hessian.

So, now you can perform these operations over multiple resolution representation, where instead of convolving using the Gaussian mask this Hessian operator itself; that means, this computation of this double derivative they are computed with a different sizes of masks. So, with increasing masks site you are obtaining the larger scales, you are equivalently performing largest smoothening of the more smoothening of the images which means of lower resolution representation of the images, and from there you try to capture the local maxima in that representation in that kind of representation of images compute the local maxima.

(Refer Slide Time: 09:43)

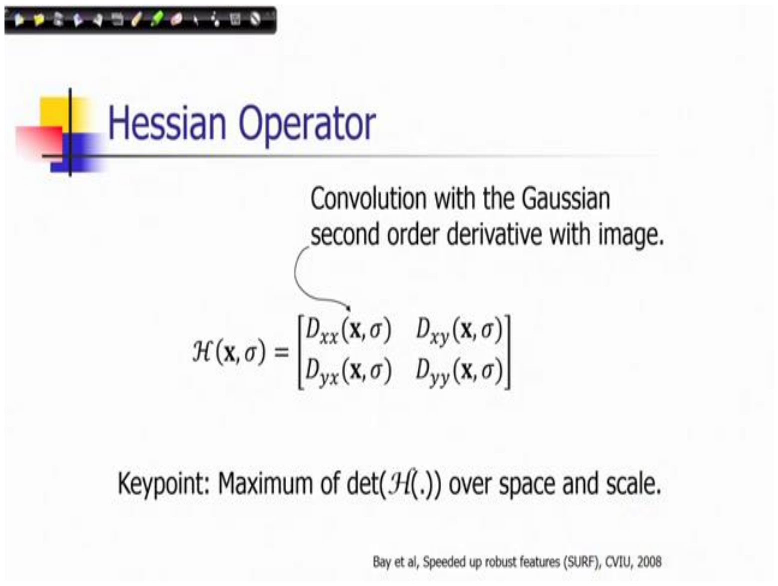


Speeded-Up Robust Features (SURF): Another descriptor

- Speeded-Up Robust Features (SURF)
  - (Bay et al. ECCV, 2006)
  - Box-type convolution filters and use of integral images to speed up the computation.
  - Use of Hessian operator for key point detection→ Local maxima of det(H).
  - Accumulate orientation corrected Haar wavelet responses.

So, we have to accumulate the orientation corrected Haar Wavelet responses in this case also. So, in the sift we have accumulated orientation corrected gradient directions and in this case the gradient computation also is carried out using Haar Wavelets which can be easily implemented by box filters. So, I will elaborate in subsequent slides.

(Refer Slide Time: 10:09)



Hessian Operator

Convolution with the Gaussian second order derivative with image.

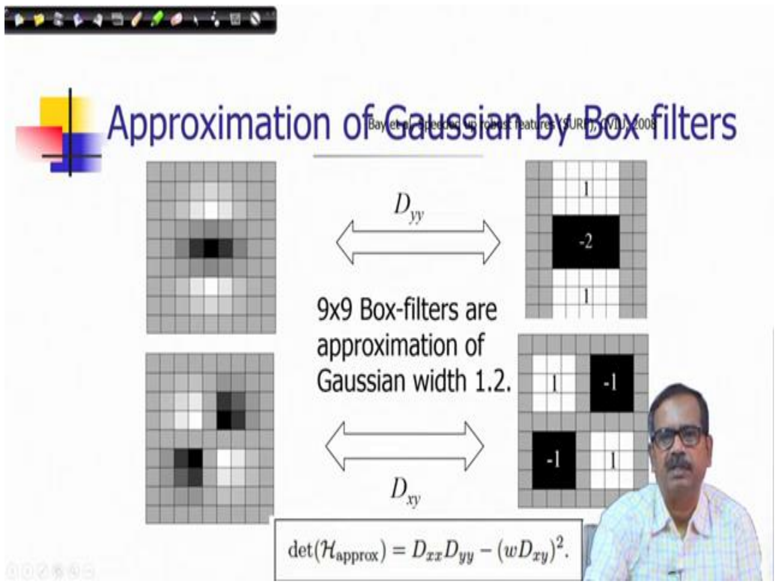
$$\mathcal{H}(\mathbf{x},\sigma) = \begin{bmatrix} D_{xx}(\mathbf{x},\sigma) & D_{xy}(\mathbf{x},\sigma) \\ D_{yx}(\mathbf{x},\sigma) & D_{yy}(\mathbf{x},\sigma) \end{bmatrix}$$

Keypoint: Maximum of det($\mathcal{H}$(.)) over space and scale.

Bay et al, Speeded up robust features (SURF), CVIU, 2008

So, this is what is the Hessian operato. So, the convolution with Gaussian second order derivative with images, but this is actually replaced by a box filter in the sharp cases. So, key point is maximum determinant of this over space and scale.

(Refer Slide Time: 10:35)



Approximation of Gaussian by Box filters

Bay et al, Speeded up robust features (SURF), CVIU, 2008

9x9 Box-filters are approximation of Gaussian width 1.2.

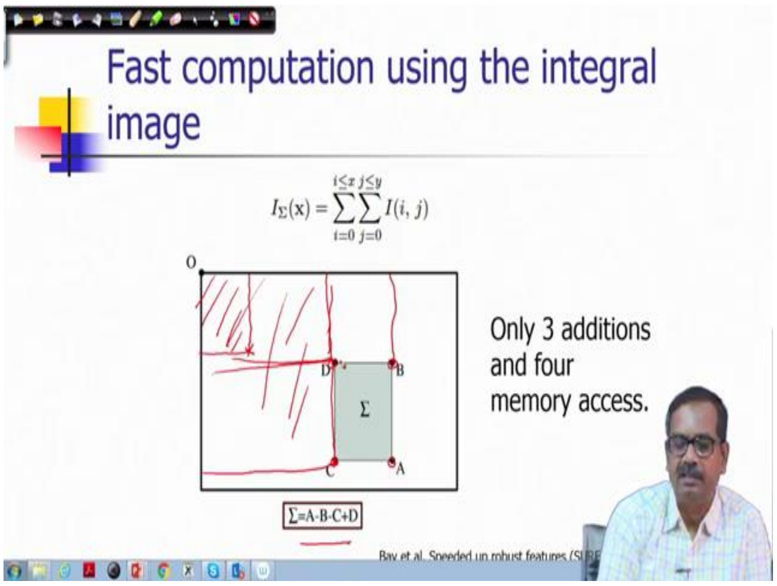
$$\det(\mathcal{H}_{\text{approx}}) = D_{xx}D_{yy} - (wD_{xy})^2.$$

So, this is approximation. You can see in this picture how the box filters they look like. So, here you can find out that this representation. So, the brighter values are shown by the values of 1 and the darker values are shown by values are -1. So, it is not only 1, there could be other integral values also.

$$\det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2$$

And here the value of ‘w’ is taken as 0.9. And particularly 9x9 box filters are an approximation of Gaussian weight 1, 2. So, I should say approximating convolutions using Gaussian masks or double derivatives of Gaussian masks.

(Refer Slide Time: 11:59)



Fast computation using the integral image

$$I_{\Sigma}(\mathrm{x}) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j)$$

Only 3 additions and four memory access.

$\boxed{\Sigma = A - B - C + D}$

Bay et al. Speeded up robust features (SURF

About the use of integral images that is also explained nicely here in this slide. Here you can what is an integral image. You can see that integral image is the cumulative sum of pixel values over a over a rectangular region.

$$I_{\Sigma}\ (x) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j)$$

$$\sum = A - B - C + D$$

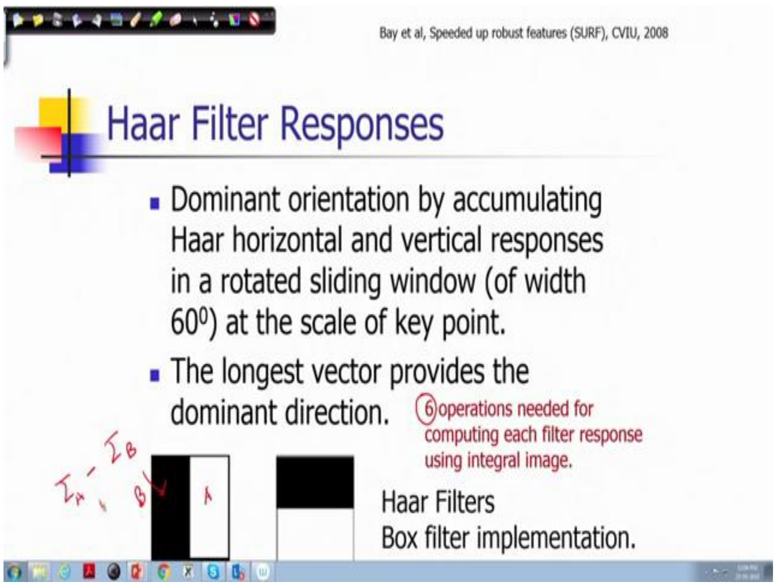
So, over this image at every point suppose I would like to compute the integral value at this point. What I have to do? I have to compute the sum of pixel values in this particular region and that value will be replaced here.

Now, these computations can be efficiently performed in one scan, because this could be accumulated. This sum could be accumulated over computations of other integer values which has been already computed. So, in one scan we can compute an integral image. Now, given that integral image I can compute sum over any rectangular zone by using these operations; that means, this is the integral image value at these points.

I need to take the sum over all this region and then subtract these values over this particular particular term, and then revalue this one, because this is twice subtracted. So, if I do that, then actually I will get the corresponding sum over this particular value.

So, when you are applying box filter over a space. So, what you are doing? You are simply performing either addition accumulating those value addition of the pixel values along that region and then in subsequently you are using this value for subtraction weighted, you can multiply it by that weight then you are using this value for subtraction or addition. You can see the box filtered implementations earlier for computing the determinant of the Hessian matrix what we discussed. And so every kind of computations of the sum of these pixel values it requires only 3 additions and 4 memory access, and that is how efficiently or it increases the speed of computations.

(Refer Slide Time: 14:57)



Bay et al, Speeded up robust features (SURF), CVIU, 2008

Haar Filter Responses

- Dominant orientation by accumulating Haar horizontal and vertical responses in a rotated sliding window (of width 60^0) at the scale of key point.
- The longest vector provides the dominant direction.

6 operations needed for computing each filter response using integral image.

Haar Filters
Box filter implementation.

And next what I said that is a computation of Hessian and finding out the corresponding key points by computing the local maxima of Hessians implement using box filters.

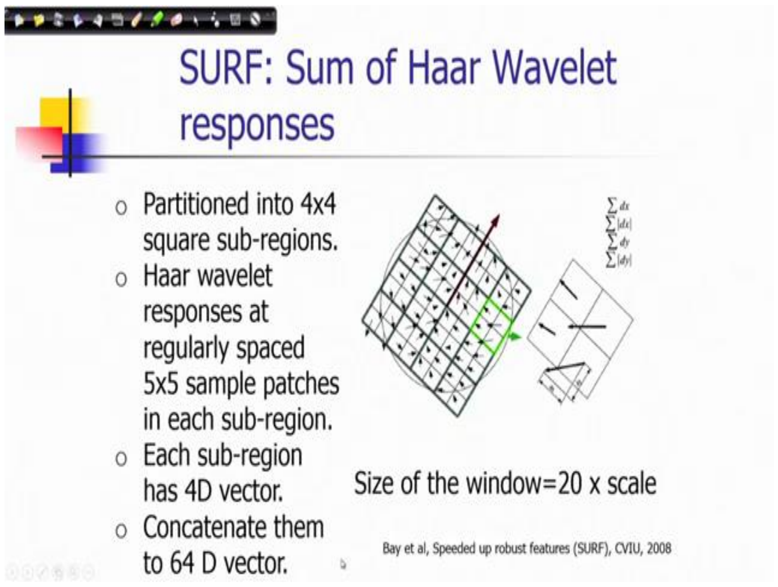
Now, for describing the key point, once again like sift, here also around its neighborhood you are accumulating the statistics of the orientations. But in this case the orientations are once again computed using Haar Wavelets. Once again this is nothing but in this case as it is computation of gradients along different scales. So, these are the two particular filters and which has been shown here and they can be implemented by box filters.

And to get the orientation, what you can do? You can rotate the image and then apply these filters and find out where you get the maximal responses. So, the longest period that would provide the dominant directions. And this could be implemented by box filters as we have discussed earlier also that white portions would be say for example, we are summing them all those pixels in the white regions and after summing all the dark pixels you are performing subtraction, which means in these computations.

For example, if I am applying this particular convolutions operations, so for summing up the pixels around this we will take 3 operations for integral image. Similarly, summing up will take 3 operations. So, total 6 operations and then you require also a subtraction, so another addition operations. But with 6 operations you can compute two particularly these two values. So, 6 operation units for computing each filter response using integral image, actually you require one more operation because after that you need to perform the subtraction.

So, you sum this value you take the summations of this value suppose this is over region A, so and then this is the region B and then you have to perform a subtraction. So, this is how you can compute these particular operations. So, instead of 6, 6 is for this 3 plus 3, but this will require another additional operation.

(Refer Slide Time: 17:43)



SURF: Sum of Haar Wavelet responses

- Partitioned into 4x4 square sub-regions.
- Haar wavelet responses at regularly spaced 5x5 sample patches in each sub-region.
- Each sub-region has 4D vector.
- Concatenate them to 64 D vector.

$\sum dx$
$\sum |dx|$
$\sum dy$
$\sum |dy|$

Size of the window=20 x scale

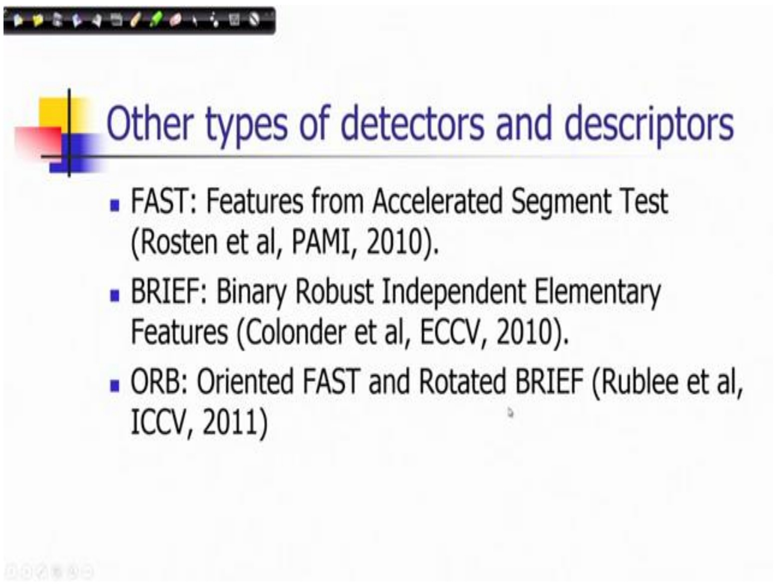
Bay et al, Speeded up robust features (SURF), CVIU, 2008

And to get the final descriptor, what we do? We do not know collect like the meaning of histogram, so I did in this case of sift. So, instead of that in this case after performing these orientation corrections we partition these square sub regions into 4x4, partition sub regions into 4x4 quiet sub regions and in each region we are computing this particular quantities. That means, after performing the Haar Wavelets, so we are taking the summation of the convert responses and also the summation of the magnitudes of the converted responses of Haar Wavelets along the x direction. Similarly, those two along y directions.

So, as you can see that each region each cell of 4x4 cell will give me a 4x4 dimensional vector, and there would be and size of the window is twenty cross scale that is the size of the window and since there are 4 cross 4 square regions. So, each region will give me a 4 dimensional vector.

So, you will get, so the it is regularly spaced 5x5 sample patches in each sub region and then each sub region has 4 dimensional vector, and if you concatenate them then it would be a 64 dimensional vector and that is how the surf response is given. So, you can see the reference of this paper is given here for the details you should look at this paper which you can know read in details and you can get all this description.

(Refer Slide Time: 19:44)



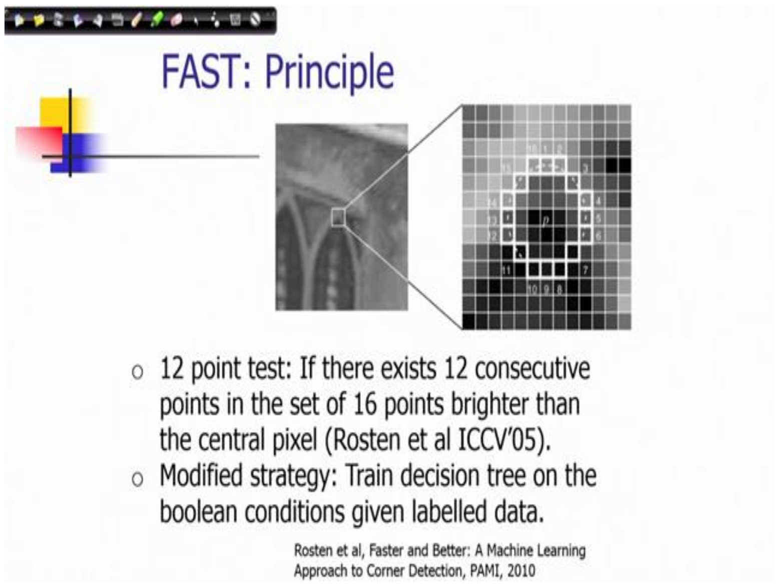
Other types of detectors and descriptors

- FAST: Features from Accelerated Segment Test (Rosten et al, PAMI, 2010).
- BRIEF: Binary Robust Independent Elementary Features (Colonder et al, ECCV, 2010).
- ORB: Oriented FAST and Rotated BRIEF (Rublee et al, ICCV, 2011)

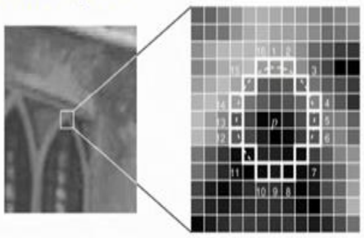
So, we discussed about two such descriptors like sift and surf, but still they are computationally quite intensive. So, there are different other descriptors which are proposed later and they also quiet becoming popular in various applications. Particularly, you require first computations in real time applications. So, one of such detectors is called FAST detector. By the name FAST it is implying that it is very it computes first, the detection of feature point. So, but the acronym is features from accelerated segment test. I will have a BRIEF description of that particular method.

And also there are different descriptors like BRIEF or binary robust independent elementary features or ORB which makes the description rotational invariant by considering oriented FAST detection. So, FAST is not a rotation invariant detection. But in ORB we know that limitation has been overcome by performing some operations which I will be also discussing. And also the rotated BRIEF, so BRIEF is also not orientation independent descriptor. So, in ORB those are taken care of.

(Refer Slide Time: 21:10)



FAST: Principle



- 12 point test: If there exists 12 consecutive points in the set of 16 points brighter than the central pixel (Rosten et al ICCV'05).
- Modified strategy: Train decision tree on the boolean conditions given labelled data.

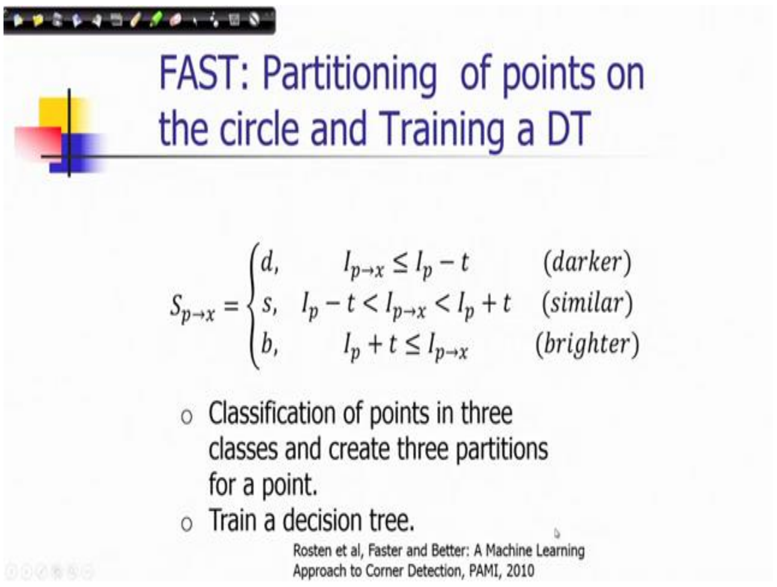
Rosten et al, Faster and Better: A Machine Learning Approach to Corner Detection, PAMI, 2010

So, this is the principle of detection of a key point or feature point in FAST. and there is a test which is called 12 point test and where it considers much defined locations around a pixel. And those are shown in the diagram which are the locations shown by the square thick squared edges. So, you can say that the size of the mask in this case is about is 7x7 mask and there are location centering this particular pixel. So, this is the central pixel and this location, so this is 1 2 3.

So, if I consider these are the locations say 1 2 3 like this, so there are in fact 16 such, no, locations which are shown here. But out of them if they are exist 12 consecutive points which are brighter than the central pixel then we consider that point is an interesting point which is a key point, that is the principle. So, it has been shown that this works well for detecting various key points.

So, there are also modification of these strategies which improves this performance. Now, we can train decision tree on the Boolean conditions given level data. So, the details you can find those details in this particular paper which has been shown here. It is a paper published in transaction, transactions of pattern analysis and machine intelligence in 2010 and is written by Rosten et al. So, you can look at this paper and get the details.

(Refer Slide Time: 23:20)



FAST: Partitioning of points on the circle and Training a DT

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t \quad (darker) \\ s, & I_p - t < I_{p\to x} < I_p + t \quad (similar) \\ b, & I_p + t \le I_{p\to x} \quad (brighter) \end{cases}$$

- Classification of points in three classes and create three partitions for a point.
- Train a decision tree.

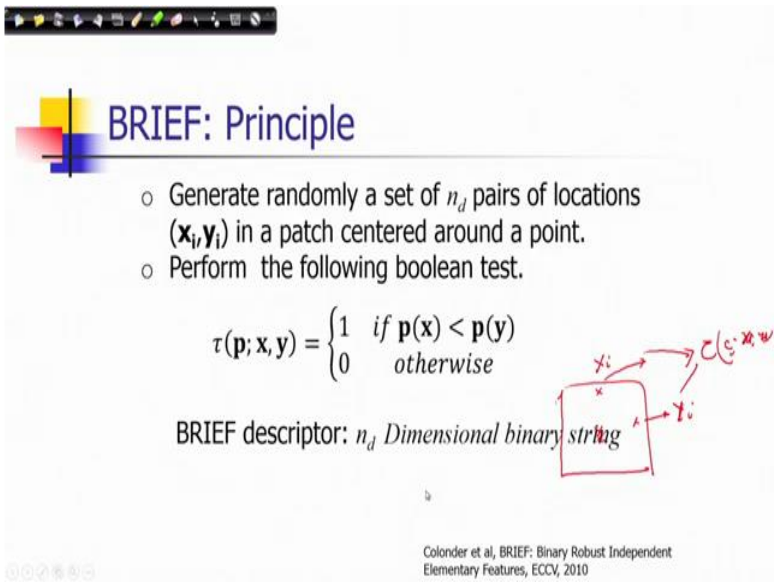
Rosten et al, Faster and Better: A Machine Learning Approach to Corner Detection, PAMI, 2010

So, just a little more elaborations about this particular point. So, there are even for the decision trees that partitioning of points on this circle and you can train them as a decision tree. So, you can characterize them as a point which is in the category of darker and a point which are similar with respect to center of pixel, either it is darker than the central pixel or almost similar or it is brighter. So, then it is a trinary logic.

So that means, there are 3 steps for every location, every point which are shown out of those 16 points. And then based on them you can train a decision tree based on those values. And given training samples of interesting key points and their neighborhoods, use that decision tree later on to identify key points. So, the problem is the classification of points in 3 classes and create 3 partitions for a point and then you can train a decision tree.

$$S_{p\to x} = \begin{Bmatrix} d, I_{p\to x} \le \ I_p - t \ (darker) \\ s, I_p - t < \ I_{p\to x} < \ I_p + t \ (similar) \\ b, I_p + t < \ I_{p\to x} \ (brighter) \end{Bmatrix}$$

(Refer Slide Time: 24:27)



BRIEF: Principle

- Generate randomly a set of n_d pairs of locations $(\mathbf{x}_i, \mathbf{y}_i)$ in a patch centered around a point.
- Perform the following boolean test.

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) = \begin{cases} 1 & if\ \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & otherwise \end{cases}$$

BRIEF descriptor: n_d *Dimensional binary string*

Colonder et al, BRIEF: Binary Robust Independent Elementary Features, ECCV, 2010

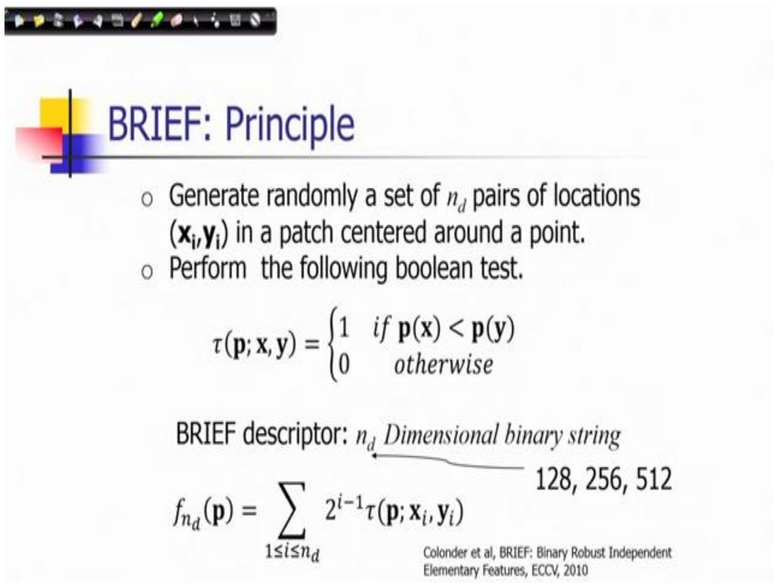
So, now let me discuss about a descriptor around a key point. So, the previous one is a replacement of the detector operator like Harris operator or difference of Gaussian operations, max local maximum of difference of Gaussian, local maxima of Laplacian measure. So, first replaces them and as you can see the computation is very simple based on those heuristics and those principles.

Now, let us see that how a descriptor also could be computed without too much complex computation. So, in this case this is a principle that we considered randomly a set of n_d pairs of locations which is denoted here a set(x_i, y_i) that is i^{th} point in that set in a patch centered on a point. Actually, x_i is not x coordinate, $x_i$is the point, $y_i$is the point and they are coupled together.

$$\tau(p; x, y) = \begin{Bmatrix} 1, if\ p(x) < p(y) \\ 0, otherwise \end{Bmatrix}$$

So, suppose you have a pixel here, some central pixel ‘c’ and around its neighborhood you have you have randomly defined certain pairing a point. So, these are the positions and then you are getting a binary value 1, right. So, you get a binary string. So, there are n_d such pairs, so you get a binary string of lengthn_d. In fact, that is what is supposed to your future descriptor.

(Refer Slide Time: 26:32)



BRIEF: Principle

- Generate randomly a set of n_d pairs of locations $(\mathbf{x}_i, \mathbf{y}_i)$ in a patch centered around a point.
- Perform the following boolean test.

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) = \begin{cases} 1 & if\ \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & otherwise \end{cases}$$

BRIEF descriptor: n_d *Dimensional binary string*

128, 256, 512

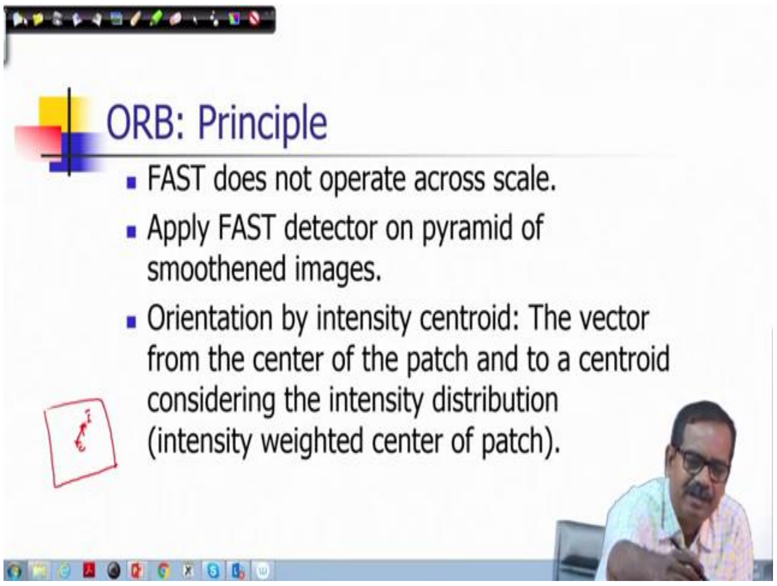
$$f_{n_d}(\mathbf{p}) = \sum_{1 \le i \le n_d} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}_i, \mathbf{y}_i)$$

Colonder et al, BRIEF: Binary Robust Independent Elementary Features, ECCV, 2010

$$f_{n_d}(p) = \sum_{1 \le i \le n_d} 2^{i-1} \tau(p; x_i, y_i)$$

So, it is n dimensional binary string and as you can see this computation is very fast and it has been found that this is also giving a feature descriptor around key point which is quite unique. And you can consider a value of this point also, you can you can know it since it is a binary representation, so you can also convert this into a decimal value. So, the typical dimensions which are used to in experiments like 128, 256, 512, you can see that a very large dimension.

(Refer Slide Time: 27:10)



ORB: Principle

- FAST does not operate across scale.
- Apply FAST detector on pyramid of smoothened images.
- Orientation by intensity centroid: The vector from the center of the patch and to a centroid considering the intensity distribution (intensity weighted center of patch).

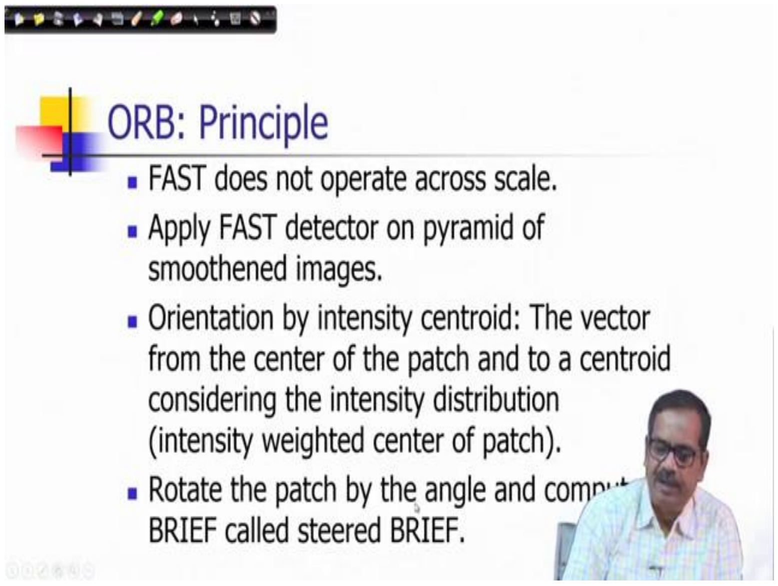
But the problem is that as we can see that now there is no concept of making it rotational invariant, there is no concept of making it scale invariant. Even the FAST also there is no concept of rotational invariant or scaling variant. So, in ORB that is another descriptor this these are taken care of.

So, again the multi resolution images are representation is used and orientation compensation has been done in a very simple and tricky way that now, you consider an intensity centroid. So, that is the characteristic. So, what is an intensity centroid? Say, you have a key point which is given by c, now this is a geometric center.

Now, if I consider the centroid of the intensity values; that means, a pixel which contains the average of the intensity values which is close to the average of the intensity values then that is intensity weighted center of patch it is not average. So, we are considering the location where intensity is acting like a weight and taking the intensity weighted central location. So, that is the different point. And this gives me an orientation. It is something like it will try to simulate the gradient direction. So, this is the orientation.

So, now, you describe all those, you perform all those operations by performing rotations by aligning your reference access to this orientation and you can make it this rotational invariant in this way. So, rotate the patch by the angle and compute degree, and this is called actually steered BRIEF.

(Refer Slide Time: 28:56)



ORB: Principle

- FAST does not operate across scale.
- Apply FAST detector on pyramid of smoothened images.
- Orientation by intensity centroid: The vector from the center of the patch and to a centroid considering the intensity distribution (intensity weighted center of patch).
- Rotate the patch by the angle and comp BRIEF called steered BRIEF.

So, let me stop here at this point, where we have discussed know various descriptors, I mean other than surf and sift which are quite computationally intrinsic, but in there are also additional descriptors like BRIEF, ORB and also a FAST detector which acronym is also FAST and they are computationally efficient.

So, next we will we will continue this discussion once again just to understand that how these descriptors could be used in matching a pairs of key points, and also there are other kinds of descriptors which are also useful in the in various other bit processing of images on analysis of images that we will also discuss. So, let us stop at this point for this particular lecture.

Thank you very much.

Keywords: SIFT, SURF, FAST, BRIEF