

Deep Learning for Computer Vision
Prof. Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad
Lecture 19
Pyramid Matching

Continuing with the previous lecture, we will now go from matching kernels to matching kernels for image pyramids. For example, having a multi-resolution pyramid of images and being able to use that idea to develop matching kernels.

(Refer Slide Time: 00:34)



The screenshot shows a presentation slide with a grey header bar at the top. Below the header, there is a grey box containing the text "Acknowledgements" followed by a bullet point: "Most of this lecture's slides are based on lectures of **Deep Learning for Vision** course taught by Prof Yannis Avrithis at Inria Rennes-Bretagne Atlantique". To the right of the slide, there are two logos: the NPTEL logo (National Programme on Technology Enhanced Learning) and the IIT Hyderabad logo (Indian Institute of Technology Hyderabad). At the bottom of the slide, there is a footer bar with the text "Vineeth N B (IIT-H) | 3.5 Pyramid Matching | 2/2". A small video inset of the professor is visible in the bottom right corner of the slide.

The slides have once again borrowed from the lectures of Professor Avrithis at Inria Rennes.

(Refer Slide Time: 00:41)

Recall: Descriptor Matching

- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, $X_c = \{x \in X : q(x) = c\}$ where q maps vector x to its nearest centroid, bag-of-words similarity on C is given by:

$$s_{BoW}(X, Y) \propto \sum_{c \in C} w_c |X_c| |Y_c| = \sum_{c \in C} w_c \sum_{x \in X_c} \sum_{y \in Y_c} 1 \quad (1)$$

- More general form:

$$K(X, Y) := \gamma(X)\gamma(Y) \sum_{c \in C} w_c M(X_c, Y_c)$$

where M is a within-cell matching function, and $\gamma(X)$ serves for normalization



Descriptor matching, as we just saw in the earlier lecture can be given by, you have X_c and similar Y_c for the features that belong to a particular visual word in images X and image Y . Then a matching kernel for something like bag of words could be given by summation over the same cluster centroids just counting the number of features belonging to it. You could also include some weighting factor for each of these summations if required.

And a more general form that we saw last time was what you see below here, which is

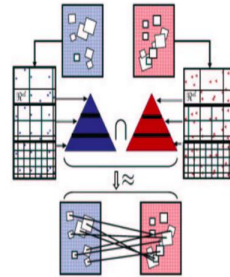
$$K(X, Y) = \gamma(X)\gamma(Y) \sum_{c \in C} W_c M(X_c, Y_c)$$

W_c is a weight that we are introducing, which we can choose to use or not and M of X_c Y_c , where M is the matching function.

(Refer Slide Time: 01:45)

Going Beyond Single-level Matching: Pyramid Match Kernel (PMK)¹

- **Pyramid matching**: an efficient method that maps unordered feature sets to multi-resolution histograms
- Computes a weighted histogram intersection to find implicit correspondences based on finest resolution histogram cell where a matched pair first appears
- Approximates similarity measured by optimal correspondences between feature sets of unequal cardinality



¹Grauman and Darrell, The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features, IEEE ICCV 2005, Vol 2, pp. 1458–1465

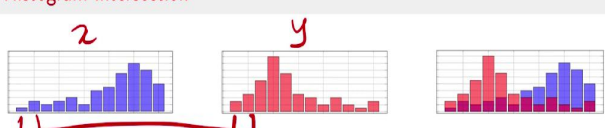


Now, we will talk about going beyond single level matching and matching at the level of pyramids and we will describe a seminal work in this context known as pyramid match kernels. So pyramid matching is an efficient method that maps unordered feature sets, which is what each image is, each image is an unordered set of features. We are going to convert that into multi-resolution histograms and then do matching using weighted multi-resolution histograms.

So we ideally can start with the finest resolution histogram cell where a matched pair first appears, and then we keep merging histograms as we go up the pyramid in this particular context. And the work has a very nice interpretation where it can be shown that it approximates a similarity in a partial matching setting, where if you had only a partial set of features in one image, match a feed, set of features in another image, pyramid match kernel approximates that optimal partial matching between those two images. For more details, I would also recommend you to read this paper called pyramid match kernel. It is written very well and explain some of these ideas in detail if you are interested in knowing more.

(Refer Slide Time: 03:22)

Histogram Intersection²





Given two histograms x, y of b bins each, their **histogram intersection** is:

$$\kappa_{HI}(x, y) = \sum_{i=1}^b \min(x_i, y_i)$$

This is related to L_1 distance as:

$$\|x - y\|_1 = \|x\|_1 + \|y\|_1 - 2\kappa_{HI}(x, y)$$

²Swain and Ballard, Color Indexing, IJCV 1991, pp 11-32
Vineeth N B (IIT-H) 33.5 Pyramid Matching 5/2



Let us start with defining histogram intersection, because we are going to define histograms in both images. Obviously, we are going to define them at multiple levels, but let us just talk about how do you match histograms in this context. So if you had two histograms x and y of b bins each, so let us say this is x , and this is y , both are histograms with b bins each.

We define the histogram intersection as minimum of x_i, y_i , one element of the histogram in both of these images and sum these up over all the b bins. So you take the first bin of histogram, the first bin of the second one, take the minimum value, take the minimum value of the next bin in the histogram and add them all. That is what we define as the histogram intersection.

Interestingly, you can show that this notion of histogram intersection, which we define as κ_{HI} has a relation to L_1 distance. We are not going to prove it here, but probably leave it as an exercise for you to look at. You can see that the L_1 distance between two vectors, x and y can be given by

$$\|x - y\|_1 = \|x\|_1 + \|y\|_1 - 2\kappa_{HI}(x, y)$$

Try it out for yourself. Take a few exits, take a couple of examples of x and y, you will actually see that it was in practice. Please try proving this also if you can. This is an interesting exercise for you to work out, but you can show that this histogram distance is related to the L1 distance. Remember L1 distance is the sum of absolute values of that vector.



(Refer Slide Time: 05:16)

Pyramid Match Kernel (PMK)³

Weighted sum of histogram intersections at different levels of two images approximates their optimal pairwise matching

$X = \{x_1, \dots, x_n\}, x_i \in \mathbb{R}^d$
 $Y = \{y_1, \dots, y_m\}, y_j \in \mathbb{R}^d$

³Grauman and Darrell, The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features, IEEE ICCV 2005, Vol 2, pp. 1458-1465
Vineeth N B (IIT-H) §3.5 Pyramid Matching 6/2

Let us come back to the pyramid match kernel now. So we said that the pyramid match kernel does a weighted sum of histogram intersections at different levels of two images, and it approximates optimal pairwise matching. So we first conceptually talk about it and then we will give a concrete example and go how it is done. So if you had these two images of the same object from different poses, different view angles, you could have, once again, you have you extract key points and those key points could be these laying in \mathbb{R}^d . You have a similar set of features lying in \mathbb{R}^d for the second image.

So you now, you have that entire feature space that you divide into a grid for instance. And now you are going to count how many of the features in one image occur in each of that grid of the feature space that is going to define a histogram. You match the histogram at that level. Then collapse the grid and merge regions in your grid in \mathbb{R}^d in your D dimensional vector assuming this size of the descriptor corresponding to the feature, your match at that level, so on and so forth.

And one intuition here is you want to give a higher weight to matches at a final level and a lower weight to matches at a higher level where the histogram bins maybe merged. We will give a concrete example and walk over this idea.

(Refer Slide Time: 06:50)

Pyramid Match Kernel: Method

• 1-D point sets X, Y on grid of size 1

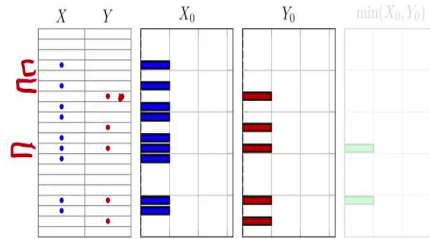
Vineeth N B (IT-H) 3.5 Pyramid Matching 7/2

Let us consider now that you have a set of features, an unordered set of features in image X , which is given by these blue points, a similar unordered set of features in image Y , even by the red points. So remember, these are points, these are descriptors of those features lying in R power d and you are going to bin them into a very fine bin of features in that space.

So it is possible that this blue point was lying in this bin, this blue point was lying in this bin and so on and so forth. You are just binning that entire R power d region into different bins and you placing each key point occurring in each image into one of those bins based on the descriptor values. Now, you have 1-D point it is X, Y on grid of size 1. We are going to call it size 1. This is the finest resolution.

(Refer Slide Time: 07:51)

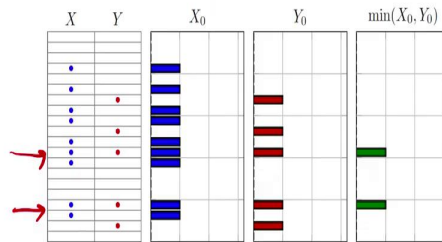
Pyramid Match Kernel: Method



o 1-D point sets X, Y on grid of size 1 - level 0 histograms



Pyramid Match Kernel: Method



o 1-D point sets X, Y on grid of size 1 - level 0 histograms - intersection



So now we define histograms. So your level zero histograms are going to be this particular bin in your R^d grid as one feature. This particular bin in R^d in X has one feature. This particular bin in R^d has one feature in image X and one feature in image Y , so on and so forth. So you can construct your histogram. Obviously, it is possible that you could have one more feature here of Y in the same bin, but at the first level, we create these bins in such a way or you can always define bins at a very fine level. That is up, you create these bins in such a way that there is only one feature in each

of the bins. We will obviously merge these as we go to be able to combine them in a more effective manner.

So based on these histograms, when you try to match them, remember our histogram intersection is going to be the mean of each element. So you are going to be left with the intersection, which is simply one value here for this bin and one value here for this bin, all of the other bins have one of the elements in X or Y to be zero, which means they will get removed.

(Refer Slide Time: 09:09)

Pyramid Match Kernel: Method

The diagram illustrates the intersection of two 1-D point sets X and Y on a grid of size 1. It shows four histograms: X, Y, X₀, and Y₀. The intersection is shown as min(X₀, Y₀).

- 1-D point sets X, Y on grid of size 1 - level 0 histograms - intersection
- 2 matches weighted by 1
- Total similarity score: $2 \times 1 = 2$

Vineeth N B (IIT-H) 3.5 Pyramid Matching 7/2

So you have two matches now between images X and Y and you are going to weight them by a value 1. So your total similarity score now is going to be 2 into 1, which is 2.

(Refer Slide Time: 09:24)

Pyramid Match Kernel: Method

- 1-D point sets X, Y on grid of size 2 - level 1 histograms - intersection
- (2 matches weighted by 1) + (2 weighted by $\frac{1}{2}$)
- Total similarity score: $2 \times 1 + 2 \times \frac{1}{2} = 3$

Vineeth N B (IIT-H) 3.5 Pyramid Matching 7/2

Now we are going to merge your histogram bins. Originally, if you had say about 20 histogram bins, you need to merge every consecutive, every contiguous one and make them into 10 bins. And now you see that it is possible that there are two features in image X that belong to the same bin and so on and so forth.

So now we construct what are known as level 1 histograms where we count the number of features in each of these merged bins in image X and image Y. You see that there are two occurrences of features in this bin. Similarly, there are two occurrences of features in this bin in image X, but image Y has just one feature still in each bin. So based on that, we build the histogram for image X, build the histogram for image Y.

And now you compute the intersection of these two histograms and you find that there are four matches. But you do not count all the four matches, you count how many new matches are added. So we are only going to look at how many new matches are added by matching these histograms, which is going to be we had two matches earlier, we have four matches now, the new matches would be two.

So now you consider those new matches, you weight them by half. Why half, remember a match at a closer level is given lesser weight than a match at a finer level, because the finer level match means a closer match. So you take these two new matches weighted

them by half and now your similarity score becomes 2 into 1 from the earlier slide plus 2 into 1/2, which is totally going to be 3.

(Refer Slide Time: 11:19)

Pyramid Match Kernel: Method

X Y X_2 Y_2 $\min(X_2, Y_2)$

- 1-D point sets X, Y on grid of size 4 - level 2 histograms - intersection
- (2 matches weighted by 1) + (2 weighted by $\frac{1}{2}$) + (1 weighted by $\frac{1}{4}$)
- Total similarity score: $2 \times 1 + 2 \times \frac{1}{2} + 1 \times \frac{1}{4} = 3.25$

Handwritten notes: 1 , $+2 = 5$, $+2$, $\text{New matches} = 1$

Vineth N B (IIT-H) 33.5 Pyramid Matching 7/2

Now, we continue this process, you now make your histogram bins just five in number, which means the number of features that you are going to have in each bin is going to increase. You can now have three features in this bin in image X, so on and so forth. Once again, you can get the histogram for X, the histogram for Y, you compute the intersection, which is now going to give you the number of matches to be 1 plus 2 plus 2, which is going to be 5, but you already had four matches in the previous level. So, the number of new matches is going to be just one, so the new match here is going to be just one.

So the similarity score now is going to be given by $2 \times 1 + 2 \times \frac{1}{2} + 1 \times \frac{1}{4}$, because you are reducing the weight even further when you go to an even higher course level. So your total similarity score is 2 plus 2 into 1/2 plus 1 into 1/4 which is going to be 3.25.

(Refer Slide Time: 12:29)

Pyramid Match Kernel



- Given a set $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in \mathbb{R}^d on a regular grid of side length 2^i
 - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has at most one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin
- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, their **pyramid match** is:


$$K_{\Delta}(X, Y) = \gamma(X)\gamma(Y) \sum_{i=0}^L \frac{1}{2^i} \left(\underbrace{\kappa_{HI}(X_i, Y_i)}_{\text{Matches at this level}} - \underbrace{\kappa_{HI}(X_{i-1}, Y_{i-1})}_{\text{Matches at previous level}} \right) \quad (2)$$

$i=1 \Rightarrow \frac{1}{2}(\kappa(x_1, y_1) - \kappa(x_0, y_0))$
 $i=2 \Rightarrow \frac{1}{4}(\kappa(x_2, y_2) - \kappa(x_1, y_1))$

where $\gamma(X)$ serves for normalization

Counts number of new pairs matched

Vineeth N B (IIT-H)
§3.5 Pyramid Matching
8/2


So let us try to put this together. So given as set X which consists of n different features each belonging to \mathbb{R}^d . Let us assume that the distances of those elements range between 1 and D . This just helps us build your bins for constructing the histogram. Once you know the maximum distance between elements, you can play around with your histogram bins to define them accordingly.

So, we are going to define X_i as a histogram of X in \mathbb{R}^d on a regular grid of side length 2^i . So we start with histogram at level 1, histogram at level zero, level 2, so on and so forth. Technically speaking, we are going to start i at minus 1, but at minus 1, there are no matches. It is purely for mathematical convenience as we will see in a moment. And then we keep building the number of histogram levels until $\log D$ where remember D use the maximum distances between the elements.

So now given two images with descriptors X and Y we are going to define formally the pyramid match as

$$K_{\Delta}(X, Y) = \gamma(X)\gamma(Y) \sum_{i=0}^L \frac{1}{2^i} (\kappa_{HI}(X_i, Y_i) - \kappa_{HI}(X_{i-1}, Y_{i-1}))$$

And at each level you are going to count the number of new matches. The first term counts the number of matches at this level, the second term counts the matches at the

previous level and you are going to keep building that. So at each point, this is going to refer to the number of new pairs matched.

So this difference can also be written, the summation of differences rather can also be written. It would, if you expand this, you would get a telescoping sum because you would have an i is equal to 0, you would have 1 by 2 power 0 into $\kappa_{HI}(X_0, Y_0) - \kappa_{HI}(X_{-1}, Y_{-1})$ which you ignore, that term is something that you ignore. Then you would have plus 1 by 2 into kappa of X_1 by 1 minus kappa X_0, Y_0 . So the X_0, Y_0 terms will be common between these two elements which will keep getting telescope.

So if you put them all together, you would find that the telescoping sum can be written as 1 by 2 power L into $\kappa(X_L, Y_L)$, which will be at the highest level plus all of the other terms will get, so, for example, let us take one particular example. If you take i is equal to 1 and i is equal to 2 . At i is equal to 1 , you are going to have $1/2$ for simplicity we just going to read it as $\kappa(X_1, Y_1) - \kappa(X_0, Y_0)$. And at i is equal to 2 , you are going to have $\frac{1}{4}\kappa(X_2, Y_2) - \kappa(X_1, Y_1)$.

So this $\frac{1}{2}\kappa(X_1, Y_1), \frac{1}{4}\kappa(X_1, Y_1)$ will get subtracted and you will be left with $1/4$ into kappa X_1, Y_1 and that is what you writing out here. So which means X_1, Y_1 would have only quarter left because one of them will get cancelled. So you will be left with 1 by 2 power i plus 1 kappa of X_i, Y_i . In case this is just a simplification of the telescoping sum that we see in the above equation. So this is just a mathematical representation of the example that we just saw over the last few slides.

(Refer Slide Time: 16:32)

PMK is a Positive Definite Kernel

- K_Δ can be written as a weighted sum of κ_{HI} terms, with non-negative coefficients $\frac{1}{2^i}$
- κ_{HI} can be written as a sum of min terms
- min can be written as a dot product:

x	$\phi(x)$
3	1 1 1 0 0 0 0
5	1 1 1 1 1 0 0
$\min(x, y) = 3$	1 1 1 0 0 0 0

- Therefore, so can K_Δ



Vineeth N B (IIT-H)

§3.5 Pyramid Matching

9/2



Pyramid Match Kernel

- Given a set $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in \mathbb{R}^d on a regular grid of side length 2^i
 - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has at most one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin
- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, their **pyramid match** is:

$$K_\Delta(X, Y) = \gamma(X)\gamma(Y) \sum_{i=0}^L \frac{1}{2^i} \left(\kappa_{HI}(X_i, Y_i) - \kappa_{HI}(X_{i-1}, Y_{i-1}) \right) \quad (2)$$

$i=1 \Rightarrow \frac{1}{2} (\kappa(x_1, y_1) - \kappa(x_0, y_0))$
 $i=2 \Rightarrow \frac{1}{4} (\kappa(x_2, y_2) - \kappa(x_1, y_1))$
 where $\gamma(X)$ serves for normalization
 Counts number of new pairs matched



Vineeth N B (IIT-H)

§3.5 Pyramid Matching

8/2



Now, it can be shown that this K_Δ function that we just defined actually happens to be a positive definite kernel. Remember again, if you recall your discussion of kernels in support vector machines and machine learning, you will recall that a positive definite kernel has benefits because it satisfies the Mercer's theorem and the computational efficiency increases, if your kernel satisfies this property. Let us see how that holds here.

Recall now that K_Δ is written as a weighted sum of κ_{HI} terms with non-negative coefficients. What are those non-negative coefficients, $1/2^i$. those

are non-negative coefficients. And then you have a weighted sum of different kappa HI terms. This is the terms that we are referring to. That is what K_{Δ} is. Or if you look at either of these equations it is simply a weighted sum of kappa HIs. And we also know that each of these κ_{HI} 's which is your histogram intersections is simply a min of values in each bin. So it is a summation of min terms.

Now we know that min can be written as a dot product. How, if you had a number 3, and if you had a number 5, I can write 3 as I put 1, 1, 1 for the first three values and then zero, zero, zero assuming I can go up to value 8. Similarly, for five, I have 1 in the first five indices followed by three zeros.

Now, the min of these two values, which is three, is simply a dot product between these two binary vectors, which means I can write min as a dot product and the rest of it now would fall in nicely because a sum of dot you would have min to be a dot product, the sum of min terms can also be written that way and a weighted sum of such kappa HI terms with non-negative coefficients can also be written this way, which means you can write your entire K delta as a positive definite kernel. In case there are parts that are not clear to you, please go ahead and read the pyramid match kernel paper to be able to get a better sense of this.

So, one question here is we just said here that min can be written as a dot product by writing out each of the numbers that you have there in this form. If you wrote out each of those numbers in this particular form, then min becomes a dot product. So you could ask me the question. You simply extrapolated the dot product to a sum of min terms and then sum of the min terms to a sum of kappa HI terms with non-negative coefficients and continued that as positive definite.

Then what would be the representation of the elements on which K delta is a positive kernel, what would be that embedding. For min, the embedding was writing it out in this manner, writing each number out simply as in enumerative, in an enumerative way. What would be the corresponding embedding upon which K delta becomes a positive definite kernel. To know what the embedding is Let us try to analyze this a bit more carefully.

(Refer Slide Time: 19:57)

PMK as an Embedding⁴

- There is an explicit embedding for κ_{HI} , therefore also for K_{Δ} . What could it be?
- If $|X| \leq |Y|$ and $\pi : X \rightarrow Y$ is one-to-one, then $K_{\Delta}(X, Y)$ approximates the optimal pairwise matching:

$$\max_{\pi} \sum_{x \in X} \|x - \pi(x)\|_1^{-1}$$

- This is similar to the **Earth mover's distance**:

$$\min_{\pi} \sum_{x \in X} \|x - \pi(x)\|_1$$

- But PMK is a *similarity* measure; it allows partial matching and does not penalize clutter, except for the normalization

⁴Indyk and Thaper, Fast Image Retrieval via Embeddings, WSCTV, 2003
Vineeth N B. (IIT-H) 3.5 Pyramid Matching 10/2



Pyramid Match Kernel

- Given a set $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, where distances of elements range in $[1, D]$
- Let X_i be a histogram of X in \mathbb{R}^d on a regular grid of side length 2^i
 - i ranges from -1 (a base case where there is no intersection, 0 (where each bin has at most one element), and so on to $L = \lceil \log_2 D \rceil$, where all of X is contained in a single bin
- Given two images with descriptors $X, Y \subset \mathbb{R}^d$, their **pyramid match** is:

$$K_{\Delta}(X, Y) = \gamma(X)\gamma(Y) \sum_{i=0}^L \frac{1}{2^i} \left(\kappa_{HI}(X_i, Y_i) - \kappa_{HI}(X_{i-1}, Y_{i-1}) \right) \quad (2)$$

$i=1 \Rightarrow \frac{1}{2}(\kappa(x_1, y_1) - \kappa(x_0, y_0))$
 $i=2 \Rightarrow \frac{1}{4}(\kappa(x_2, y_2) - \kappa(x_1, y_1))$
 where $\gamma(X)$ serves for normalization
 Counts number of new pairs matched



So if you had two images X and Y for convenience let us assume that X has a lesser number of features than image Y . Remember both of them are unordered set of features. It could be the other way too. This is without loss of generality. In that case, it would just be flipped. But otherwise you can assume that one is less than the other in terms of cardinality of features. And that is define a function π that takes us from image X to image Y in such a way that π is one-to-one, which means for every feature in image X , you find the closest feature in image Y .

In that case, the optimal pairwise matching would be given by, you take a feature from image X, you find the corresponding closest feature in image Y, you take the L1 distance between these two features and you are going to find the π of the function that takes you from image X to image Y which gives you the least which, sorry, which maximizes the reciprocal of this distance. Remember, reciprocal of this distance is going to give you a sense of similarity because of the reciprocal you want to find the function π which gives you the maximum such distance.

For those of you who are a bit more familiar with distance metrics, you would find that such a representation is similar to what is known as the earth mover's distance, which is given by $\min_{\pi} \sum_{x \in X} \|x - \pi(x)\|_1$. Remember that this is a distance metric, while this representation of optimal pairwise matching is a similarity measure, which is why you have max here and you have a min here. Remember that distance and similarity are complimentary ideas. If one is high, the other should be low, so on and so forth.

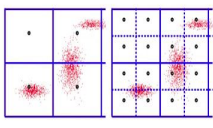
So it happens that defining X the way we did, where we defined it in terms of grid locations and histograms and so on and so forth and taking a one norm between those intersections, actually gives us the embedding. For more details of this, this could be a bit mathematically involved, but for details of this, please see this particular paper called fast image retrieval via embeddings. But the core idea that you want to take away from here is that the pyramid match kernel defines a positive definite kernel which makes it efficient because we know that a positive definite kernel that satisfies the Mercer's theorem has a certain benefit in computations using the kernel trick and also that the embedding that corresponds to the kernel comes from a, can be related to the L1 distance between these X values and this particular paper describes this in more detail.

And remember that once again pyramid match kernel is a similarity measure as any other kernel functions and it does not penalize clutter except for the normalization. By that what we mean is it is possible that many features could be congregated in a certain section of your entire R power d space and you are not going to penalize it because that would just increase the histogram intersection count in a particular bin or so on and so

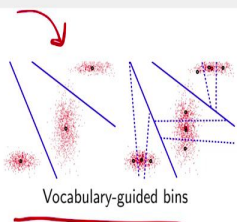
forth. There is no penalization for that. The only penalization that you could have is the normalization factor that you may be having here in your kernel definition.

(Refer Slide Time: 23:51)

PMK and Vocabulary Tree⁵





Uniform bins




Vocabulary-guided bins

- Replace regular grid with hierarchical vocabulary cells
- Compared to vocabulary tree, there is a principle in assigning cell weights
- Still, its approximation quality can suffer at high dimensions

⁵Grauman and Darrell, Approximate Correspondences in High Dimensions, NeurIPS 2007

Vineeth N B (IIT-H) 33.5 Pyramid Matching 11/2



One could extend this instead of dividing R power d into a uniform grid where you count how many features are lying in each of that R power d grid. You could also cluster all your features and now do it based on a vocabulary. So you could construct your entire histogram based on, until now in the method that we discussed, the histograms need not have been based on a vocabulary, they could have just been dividing your entire R power d into several bins and counting how many features occurred in each of those grids. But you could also consider clustering them, clustering your key points into vocabulary and then building your bins based on those cluster centers.

This would simply be an extension of the method that we have so far, where we would replace the regular grid with say hierarchical or non-hierarchical vocabulary cells. And compared to the vocabulary tree earlier at the beginning of the last lecture, we talked about how hierarchical K means can be used in bag of words. And we said that one of the concerns there is, there is no principle way of giving weights to each level in the tree. Now, in pyramid match kernel we actually have a principle way which has given by 1 by 2 power i . Even here, the approximation quality can suffer at high dimensions simply

because of the curse of dimensionality and how distance get distorted in higher dimensions.

(Refer Slide Time: 25:25)

PMK and Spatial Matching⁶

Optimal Matching

Representation

- Same idea, applied to image 2-D coordinate space for spatial matching
- Matching cost is only based on point coordinates. No appearance

⁶Grauman and Darrell, Fast Contour Matching Using Approximate Earth Mover's Distance, CVPR 2004

Vineeth N B (IIT-H) 33.5 Pyramid Matching 12/2

One could extend this idea of pyramid match kernel to do a pure special matching approach. So far, we talked about dividing. You take all the features from different images and you divide entire R^D which is the D dimensional descriptors for the features into grids and then build your histograms. But you could also build these histograms on your image space.

In this context, what you will do is, let us say you have an image such as this, there a person is performing a certain action. You could divide the image into four parts, into 16 parts and so on and so forth. And you have two different images. Now you can do matching based on histograms. How many points belong to this bin, how many points belong to the top right bin, so on and so forth. Clearly in this approach, you are only considering the coordinate locations of the features. You are not considering the descriptor or the appearance of how that feature looks at all.

But this approach could be used in trying to match say a person's position or how different a person's position was with respect to an earlier position so on and so forth. So this can be used, but has its own limitations, because in this case, you are simply counting

how many the histograms turn out to be in the spatial image space, dividing the image into parts rather than taking the descriptor of the key point and doing the histogram in the descriptor space. So you are only considering coordinates here or the geometry of the points in the image rather than how each of those key point appears.

(Refer Slide Time: 27:14)

Spatial Pyramid Matching (SPM)⁷

○ If $X^{(j)}, Y^{(j)}$ are the feature coordinates of images X, Y with descriptors assigned to visual word j ,

$$K_{SP}(X, Y) = \sum_{j=1}^k K_{\Delta}(X^{(j)}, Y^{(j)})$$

⁷Lazebnik et al, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006
Vineeth N B. (IIT-H) 3.5 Pyramid Matching 13/2

You could also combine these ideas to perform what is known as spatial pyramid matching. This was an extension of the pyramid match kernel. In this context, what you can do is you have a level zero again, very similar to pyramid match kernels, where you take a set of vocabulary, you cluster all your features into a vocabulary and then you count how many points belong to each of these cluster centers and you would get, say, histogram bins, such as these.

Now, you divided your image into four parts. And now similarly, get a histogram bin for each of these visual words for each of these segments. For the top left segment, you would once again get a histogram of three bins. For the bottom right segment, you would get a histogram of three bins, so on and so forth. So the three bins come from the vocabulary guided pyramid match kernel, where instead of dividing your descriptor space into uniform bins, you build cluster centers similar to bag of words and then you count the number of features belonging to each of those visual words.

You can once again divide the image even further. Now, you are going to get even higher number of histogram bins corresponding to each of these locations. So in this case, your kernel is going to be, you have your pyramid match kernel, but you are now going to do that for each part of the image and add them all up. So the pyramid match kernels still exists for each part of the image and then you keep doing this over different parts of the image.


(Refer Slide Time: 28:51)

Spatial Pyramid Matching (SPM)⁷

- Coupled with BoW, it is a set of joint appearance-geometry histograms
- Robust to deformation but not invariant to transformations; Applied for global scene classification

⁷Lazebnik et al, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006.

Vineeth N. B. (IIT-H)
§3.5 Pyramid Matching
13/2



So, one could look at it as a joint appearance geometry histogram. So pyramid match kernel was a pure appearance histogram because you had building the histograms in the descriptor space. We saw an example of how pyramid match kernels can be brought to special matching, which was a pure geometry histogram and spatial pyramid matching brings these two together to create what are known as appearance geometry histograms.

So these are robust to deformation, not completely invariant to transformations, but fairly robust to deformation by simply the process that you are defining, where you are considering the appearance as well as where each of these features occurred in a given image, which was not there in the pyramid match kernel at all. So this can be used for global scene classification where a different organization of objects should not distort your final result.

(Refer Slide Time: 29:55)

Hough Pyramid Matching (HPM)⁸



Fast Pyramid Matching

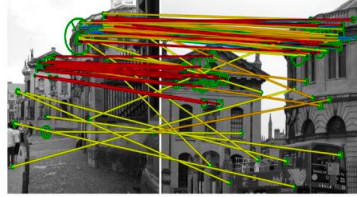
⁸Tolias and Avrithis, Speeded-up, relaxed spatial matching, ICCV 2011
Vineeth N B (IIT-H) 33.5 Pyramid Matching



A last method that we will talk about in this lecture is hough pyramid matching, which is clearly an extension to hough voting if you recall. So, in this method, the idea is, remember that in typical pyramid matching, you would take a set of features and match them to features from another image and you could do this in a fast manner by using image pyramids if you recall discussions in earlier lectures, where you first do matching at a coarse level, then to final matching at a deeper level of the pyramid and so on and so forth.

(Refer Slide Time: 30:35)

Hough Pyramid Matching (HPM)⁸



Hough Pyramid Matching

- Work with a single set of correspondences instead of two sets of features
- Determine a transformation hypothesis by a pair of features and then use histograms to collect votes in the transformation space

⁸Tolias and Avrithis, Speeded-up, relaxed spatial matching, ICCV 2011

Vineeth N B (IIT-H)

§3.5 Pyramid Matching

14/2



So you could have a bunch of correspondences which you get from matching at the level of key points. And what we are going to do now is work with these correspondences instead of two sets of unordered features. So you have a set of correspondences that you already get by doing fast pyramid matching. And remember the central idea of hough voting is each of your correspondences votes for a particular transformation or you have a hypothesis of transformation based on say the rotation angle or scale or translation and each of these correspondences votes for a particular hypothesis and we are now going to build histograms in that transformation space.

(Refer Slide Time: 31:21)

Hough Pyramid Matching

- A local feature p in image P has position $t(p)$, scale $s(p)$ and orientation $\theta(p)$ given by matrix $R(p) \in \mathbb{R}^{2 \times 2}$.

$$F(p) = \begin{pmatrix} s(p)R(p) & t(p) \\ 0 & 1 \end{pmatrix}$$

Handwritten notes in red ink show the expansion of the matrix $F(p)$ into a 3x3 form:

$$\begin{pmatrix} r \cos \theta & r \sin \theta & t_x \\ -r \sin \theta & r \cos \theta & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

Arrows indicate the mapping from the terms in the matrix to their corresponding variables: r for scale, θ for orientation, and t_x, t_y for position.



Vineeth N B (IIT-H)

3.3.5 Pyramid Matching

15/2



Let us see an example here. You could assume that a local feature P in image P has a certain scale, orientation and translation to it, position to it in this particular case, so which is given by this transformation matrix. Remember that this transformation matrix is just a different way of writing what we saw earlier, where we saw you have $r \cos \theta$ $r \sin \theta$ $-r \sin \theta$ $r \cos \theta$ t_x , t_y , zero, zero, 1, which constituents in a fine transformation where r is a scale, θ is the orientation and t_x and t_y are positions.

So this is just a concise way of writing such a matrix. So, these two zeros correspond to this zero vector here. One is there for mathematical simplicity and then this $s(p)$, $R(p)$ corresponds to the scale and orientation of that point P , which can be written as a two by two matrix and this vector t of p corresponds to the position of that particular point in the image.

(Refer Slide Time: 32:41)

Hough Pyramid Matching

- A **local feature** p in image P has position $\mathbf{t}(p)$, scale $s(p)$ and orientation $\theta(p)$ given by matrix $R(p) \in \mathbb{R}^{2 \times 2}$.

$$F(p) = \begin{pmatrix} s(p)R(p) & \mathbf{t}(p) \\ \mathbf{0}^T & 1 \end{pmatrix}$$

- A **correspondence** $c = (p, q)$ is a pair of features $p \in P, q \in Q$ of two images P, Q and determines relative similarity transformation from p to q :

$$F(c) = F(q)F(p)^{-1} = \begin{pmatrix} s(c)R(c) & \mathbf{t}(c) \\ \mathbf{0}^T & 1 \end{pmatrix}$$

with translation $\mathbf{t}(c) = \mathbf{t}(q) - s(c)R(c)\mathbf{t}(p)$, relative scale $s(c) = s(q)/s(p)$ and rotation $R(c) = R(q)R(p)^{-1}$ or $\theta(c) = \theta(q) - \theta(p)$



Assuming this is how a local feature is given to us. Then a correspondence between a pair of features $p \in P$ and $q \in Q$ can be given by, $F(c) = F(q)F(p)^{-1}$, remember $F(p)$ is one point p 's representation, similarly, $F(q)$ would be the point q 's representation in image Q and the corresponding, the correspondence between these two points is given by $[[s(c)R(c) \mathbf{t}(c)], [0 \ 1]]$. Once again this boils down to your rotation and scale matrix coming here, your translation t_x, t_y coming here, and your zero, zero, 1. Now, t_x, t_y are not just the coordinates, they are how much you moved from the coordinated image X or point P to the coordinate q in image Q .

Similarly, the scale and the rotation tells you, what is the transformation, how much did you rotate to go from image P to image Q , and how much did you zoom in or zoom out to go from image P to image Q . So t_c , so we are not going to go deeper into this, but just to complete this discussion, this t_c can be written as t_q , which is the coordinate location of Q minus $s_c R_c t_p$. Why is that so? t_q is the position in of q in image Q , t_p is the position of point p in image P and s_c, R_c says, how did you rotate p and how did you zoom p to get to a point in image Q and the difference between those two locations is going to be the actual translation t_c .



Similarly, you can define the relative zoom in or zoom out to be the scale in q divided by the scale in p and the rotation, similarly, to be given as R_q into R_p inverse or the angle is given by the orientation and image of point q in image Q minus theta of p the orientation of p in P . This is how the correspondence is given.


(Refer Slide Time: 34:52)

Hough Pyramid Matching

- The 4-DoF relative transformation represented by 4-D vector:

$$f(c) = (t(c), s(c), \theta(c))$$
- To enforce one-to-one mapping, two correspondences $c = (p, q)$ and $c' = (p', q')$ are said to be **conflicting** if they refer to the same feature on either image, i.e., $p = p'$ or $q = q'$

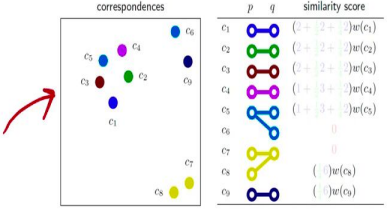



Vineth N B. (IIT-H)
33.5 Pyramid Matching
16 / 2


So, now let us come back to hough pyramid matching. So which means the transformation can be given by a 4-D vector $t(c)$ which as t_x and t_y , $s(c)$, the scaling factor, and $\theta(c)$, which is the orientation of the rotation difference. So you are going to define one more thing before we go into the hough pyramid matching, where if you had two correspondences p, q , and p', q' , we say that these two correspondences are conflicting if either p is equal to p' or q is equal to q' or rather if two points from image P match to the same point in image Q or one point from image P matches to two points in image Q , you call such correspondence to be conflicting. You are going to see how to use this when we go to the next slide.

(Refer Slide Time: 35:43)


Hough Pyramid Matching



	p	q	similarity score
c_1			$(2 + 2 + (-1))w(c_1)$
c_2			$(2 + 2 + (-2))w(c_2)$
c_3			$(2 + 2 + (-3))w(c_3)$
c_4			$(1 + 3 + (-2))w(c_4)$
c_5			$(1 + 3 + (-2))w(c_5)$
c_6			0
c_7			0
c_8			$(-1)w(c_8)$
c_9			$(-1)w(c_9)$

Correspondence c weighted by $w(c)$ based e.g. on visual word

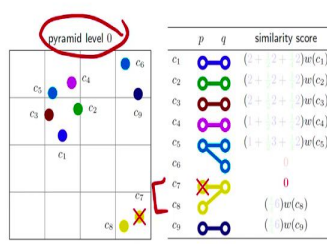
Vineeth N B (IIT-H)
§3.5 Pyramid Matching
17/2



So let us see how the hough pyramid matching actually works. So you have a set of correspondences now, which are laid out in your 4-D space, remember each correspondence as t_x, t_y, s and θ . So in this 4-D space, you are going to have each of these correspondences laid out. Now you should be able to draw similarity to the pyramid match kernel, because now you are going to be doing all your pyramid matching in this 4-D transformation space and that is why we call it hough pyramid matching. So each correspondence c is weighted by some $w(c)$ based on some say visual word. You can choose to use this or you can give a, you can have a uniform weight if you choose.

(Refer Slide Time: 36:28)


Hough Pyramid Matching



	p	q	similarity score
c_1			$(1 + 2 + 1)w(c_1)$
c_2			$(2 + 2 + 1)w(c_2)$
c_3			$(2 + 2 + 1)w(c_3)$
c_4			$(1 + 3 + 1)w(c_4)$
c_5			$(1 + 3 + 1)w(c_5)$
c_6			0
c_7			0
c_8			$(-1)w(c_8)$
c_9			$(-1)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**

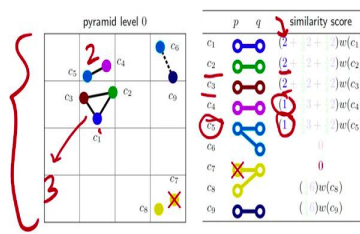
Vineeth N B (IIT-H)
§3.5 Pyramid Matching
17/2



Then at level zero, which is the first level of matching, remember where you have very granular bins. If there are conflicting correspondences in the same bin, you are going to erase them. For example, you see that c_7 and c_8 have two different points from image P matching to the same point in image Q . So you are going to remove one of them. So c_7 is removed in this case and you retain only c_8 .

(Refer Slide Time: 36:56)


Hough Pyramid Matching



	p	q	similarity score
c_1			$(2 + 2 + 1)w(c_1)$
c_2			$(2 + 2 + 1)w(c_2)$
c_3			$(2 + 2 + 1)w(c_3)$
c_4			$(1 + 3 + 1)w(c_4)$
c_5			$(1 + 3 + 1)w(c_5)$
c_6			0
c_7			0
c_8			$(-1)w(c_8)$
c_9			$(-1)w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 0 Weight 1

Vineeth N B (IIT-H)
§3.5 Pyramid Matching
17/2



Now, in each of these bins in this pyramid, remember this binning now is in the transformation space again like that 4-D space of translation, scale and rotation. So, in each of these bin b with say n_b correspondences. So, for example, this bin you have two correspondences, this bin you have three correspondences. So each correspondence groups with two others. In this case, there are three. So each correspondence groups with three others and your weight at level zero is going to be 1 very similar to how we did it for pyramid match kernel.

So, now, if you see here, you see that you have the similarity scores now, which is given by, you have for c_1 which is here, you have two new points on two new correspondences that it gets combined with. So, you put the score to be two. For c_2 , similarly, you have two new correspondences. So you put the score two. For c_3 , you have two, for c_4 you have just one new correspondence in addition to c_4 in that bin, so you put a one. For c_5 you have one new correspondence in that bin, so you put a one, so on and so forth. So you see that for c_6 , c_7 , c_8 and c_9 , there are no new correspondences. You assume now that c_9 belongs to the next bin. So it does not correspond there. So that is what you are not going to count that here. So you have, those are the similarity scores that you are going to have for this level.



(Refer Slide Time: 38:33)

Hough Pyramid Matching

	p	q	similarity score
c_1	$(2 + \frac{1}{2})$	$(2 + \frac{1}{2})$	$w(c_1)$
c_2	$(2 + \frac{1}{2})$	$(2 + \frac{1}{2})$	$w(c_2)$
c_3	$(2 + \frac{1}{2})$	$(2 + \frac{1}{2})$	$w(c_3)$
c_4	$(1 + \frac{1}{2})$	$(1 + \frac{1}{2})$	$w(c_4)$
c_5	$(1 + \frac{1}{2})$	$(1 + \frac{1}{2})$	$w(c_5)$
c_6			
c_7			0
c_8			$(-)$ $w(c_8)$
c_9			$(-)$ $w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 1 Weight $\frac{1}{2}$

Vineeth N. B. (IIT-H) §3.5 Pyramid Matching 17/2

So when you go to the next level, you merge those bins. Now you left with just two cross two gridding of that transformation space. Once again, now, you are going to erase all conflicting correspondences. Once again, here, you see that in this bin that c_7 , and c_8 are matching to the same point. So you are going to remove one of them c_7 again. Now in each of these bins you count the number of new correspondences that you have.

So c_1 now has four correspondences of which two it already got counted in the previous level. So it has two new correspondences now, which is where you are going to define for c_1 . Similarly, c_2 has two new correspondences, c_3 has two new correspondences, c_4 had one new correspondence in the previous level and four now so which has, which means it has three new correspondences. So that is the three that you see here. Similarly, c_5 has three and so on and so forth.

And the weight for this level is half. In both these levels, you do not count c_6 , because c_6 had a conflict and it was already counted as part of c_5 , those two points. So you do not count c_6 as a correspondence in this level, so which means only c_9 is left alone in this bin and hence has no new correspondences in that group and does not contribute to any new counts.




(Refer Slide Time: 40:08)

Hough Pyramid Matching

	p	q	similarity score
c_1	(2, 2)	(1, 1)	$(2 + \frac{1}{2} + \frac{1}{2})w(c_1)$
c_2	(2, 2)	(2, 2)	$(2 + \frac{1}{2} + \frac{1}{2})w(c_2)$
c_3	(2, 2)	(3, 3)	$(2 + \frac{1}{2} + \frac{1}{2})w(c_3)$
c_4	(1, 3)	(2, 2)	$(1 + \frac{1}{3} + \frac{1}{2})w(c_4)$
c_5	(1, 3)	(2, 2)	$(1 + \frac{1}{3} + \frac{1}{2})w(c_5)$
c_6			0
c_7			0
c_8	(3, 3)	(3, 3)	$(\frac{1}{3} + \frac{1}{3})w(c_8)$
c_9	(3, 3)	(3, 3)	$(\frac{1}{3} + \frac{1}{3})w(c_9)$

- Correspondence c weighted by $w(c)$, based e.g. on visual word
- Conflicting correspondences in same bin are **erased**
- In bin b with n_b correspondences, each correspondence groups with $[n_b - 1]_+$ others
- Level 2 Weight $\frac{1}{4}$

Vineeth N B (IIT-H) 3.5 Pyramid Matching 17/2

Now, when you go to the next level, where you combine everything into a single bin, you now see that each of these correspondences gets newer, gets newer correspondences. Once again, conflicting correspondences are erased, so you have c6, which is erased, similarly, c7, which is erased, and now you see that c1 has already had four correspondences, and it is going to get two new correspondences, which is why you get two here. Similarly, c2 will get two new correspondences, c3 will get two new correspondences, c4, c5 will also get two new correspondences and c8, c9 now get six new correspondences each, because they were, they did not have any correspondences in previous bins.


But the weight for each of these correspondents, since it at a higher course or level is going to be $1/4$. So now your final similarity scores can be given by 2 plus $1/2$ into 2 plus $1/4$ into 2 into any weight that you want to give for that particular correspondence w of $c1$. You can always choose to give a uniform value for w $c1$ to w $c5$ or w $c9$, so that that way does not matter or you can choose to give some weights for a given application. But this is how adding up all of these gives you the similarity scores for each of these correspondences.

(Refer Slide Time: 41:45)

Hough Pyramid Matching

o *Mode Seeking*: We are looking for regions where density is maximized in transformation space




Vineeth N B (IIT-H)
§3.5 Pyramid Matching
18 / 2


And now you try to see which of these gives you the highest density, because that is what hough voting is all about, where we try to look for regions where the density is

maximized in transformation space and then that gives you an effective match. That final bin or translation, scale and rotation value is the final match between those two points in these two images.

(Refer Slide Time: 42:16)

The screenshot shows a presentation slide with the following content:

- Hough Pyramid Matching**
- Linear in number of correspondences; no need to count inliers
- Robust to deformations and multiple matching surfaces, invariant to transformations
- Only applies to same instance matching

At the bottom of the slide, there is a footer with the text: "Vineeth N B (IT-H) 3.5 Pyramid Matching 19/2". To the right of the footer is a small video inset showing a man speaking.

So hough pyramid matching is linear in number of correspondences, which means it makes things easier and like pyramid match, where you may have to match several sets of points, you are only talking about correspondences now assuming that points have already been matched. So there is no need to count inliers here. Every correspondence can just vote and you can just take whichever one has the highest density. And it is robust deformations and multiple matching surfaces and reasonably invariant to transformations because of the way of voting proceeds.

But one important limitation here is that it only applies to same instance matching. So, once again, if you had say a bird in an image, and two such birds in the second image, you would not be able to do a matching because one bird could have match to any of these two birds in the second image, which means the transformation is to be very different and one of those could cause a problem for you in your final result.

(Refer Slide Time: 43:21)



The screenshot shows a presentation slide with a grey header bar containing the text "Homework Readings". Below the header, there are three main sections: "Homework", "Readings", and "Other Resources".

- Homework**: A red heading.
- Readings**: A grey box containing a bullet point: "Chapter 16.1.4, Forsyth and Ponce, *Computer Vision: A Modern Approach (2nd ed.)*".
- Other Resources**: A grey box containing a bullet point: "[Pyramid Match Kernel project page](#)". A red underline is drawn under this link.

On the right side of the slide, there are two logos: the NPTEL logo (National Programme on Technology Enhanced Learning) and the IIT-H logo (Indian Institute of Technology Hyderabad). At the bottom of the slide, there is a footer bar with the text "Vineeth N B (IIT-H)", "§3.5 Pyramid Matching", and "20 / 2". A small video inset of a man is visible in the bottom right corner of the slide.

Your homework for this lecture is chapter 16.1.4 in Forsyth and Ponce book and the pyramid match kernel has, also has a nice project page with more details that results with code, with the link to the paper two if you are interested in knowing any more about these methods.