CS4495 Fall 2014 — Computer Vision
Problem Set 7: Motion History Images

Officially due Friday, December 5
Accepted for full credit until **Sunday**, December 7 - 11:55pm

In this Problem Set you will implement Motion History Images. First you'll implement a simple background subtraction method (frame differencing or media/averaging). That will give you a motion signal to analyze over time.

To test this we've recorded 3 people doing 3 trials of 3 different actions. The videos are saved as separate AVI files names PS7AxPyTz.avi where 'x' is the action number. 'y' is the person number and 'z' is the "trial" number. The videos are stored in the directory `http://www.cc.gatech.edu/~afb/classes/CS4495-Fall2014/ProblemSets/PS7/Videos` .

As a reminder as to what you hand in: A Zip file that has

1. Images (either as JPG or PNG or some other easy to recognize format) clearly labeled using the convention PS ⟨number⟩–⟨question number⟩–⟨question sub⟩–counter.jpg
2. Code you used for each question. It should be clear how to run your code to generate the results. Code should be in different folders for each main part with names like PS1-1-code. For some parts – especially if using Matlab – the entire code might be just a few lines.
3. Finally a PDF file that shows all the results you need for the problem set. This will include the images appropriately labeled so it is clear which section they are for and the small number of written responses necessary to answer some of the questions. Also, for each main section, if it is not obvious how to run your code please provide brief but clear instructions. **If there is no Zip file you will lose at least 50%!**

1 Frame differenced MHI

To get images that have only the moving person we can do two things. One way is to extract the person from the background and use the entire person as a signal. We have actually found that to be more robust than the frame differencing method but requires having a background image to remove. Here we just do frame differencing.

To compute the frame differenced MHI you first need to compute the frame difference sequence. Call this the binary image $B_t(x, y)$ which is defined simply as:

$$B_t(x, y, t) = \begin{cases} 1 & \text{if } |I_t(x, y) - I_{t-1}(x, y)| \geq \theta \\ 0 & \text{if } otherwise \end{cases}$$

θ needs to be chosen such that it tends to see motion in the right places and not too much noise. To get a cleaner image you should try blurring (perhaps significantly) the original imagery. You might

want to "clean up" the binary images using a morphological OPEN operator (see the IMOPEN in Matlab) to remove noise.

1.1 Create an binary sequence from the PS7A1P1T1.avi and output binary frames for t = 10, 20, 30.

Output: Code and the three binary images.

Given the sequence B_t we can construct the MHIs. MHI image M at time$t$ is defined as:

$$M_\tau(x,y,t) = \begin{cases} \tau & \text{if } B_t(x,y) = 1 \\ \max(\ M_\tau(x,y,t-1) - 1,\ 0\ ) & \text{if } B_t(x,y) = 0 \end{cases}$$

τ needs to be chosen such that it captures the full extent of the action. If you use a different τ for each action, you should scale your MHI image by a value to make the maximum the same for all MHIs, e.g. the MHI would range in value from 0.0 to 1.0.

1.2 Create an MHI $M_\tau(x,y,t)$ for each of the three actions. Of course you'll have to pick a τ for each action and you'll also need to choose a t that "shows the action best" - probably the t where the action just ends. That is, if a sequence is 60 frames, you might select to show the MHI at time t=45 if the action ends there. And if you had $\tau = 20$, the MHI for the action would be constructed from frames 26 through 45. You can select the best t frame by hand as well as select which of the nine sequences for each action (3 subjects, 3 trials each) you use to generate these representative MHIs.

Output: Code and the three MHI images, one for each action. Say what τ you're using for each.

2 Recognition using MHIs

Now we have to characterize the MHI. In class we showed the Hu moments, though the equations we showed did not include the scale invariance. For this problem set we don't need rotation invariants so we're just going to try both the unscaled and scaled central moments μ_{pq} and η_{pq}. The equations for μ_{pq} are defined with respect to the regular moments:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

Given this definition, the average x and y are:

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}}$$

and the central moments are defined as:

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x,y)$$

Under this definition, note that μ_{10} and μ_{01} are both zero. To achieve *scale* invariance we define the scale invariant moments:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{(1+\frac{p+q}{2})}}$$

You're going to compute all the moments for $\langle pq \rangle \in \{20, 02, 12, 21, 22, 30, 03\}$. And, you'll compute this for both the MHI and the thresholded MHI where any non-zero value is set to 1 (this is the MEI from lecture). This gives you 14 numbers to describe the MHI. You might want to normalize the intensities of the MHI by scaling the values of the MHI pixels so that the maximum value is some fixed number. Also, for each of the given sequences you'll have to pick a time t that is the MHI to represent that action. Again, these can be picked by hand.

With this description you can now do nearest neighbor recognition in a "leave one out cross validation" recognition test. In this case we'll do the easiest case: simply test the MHI description for each video against the others. The "nearest" one is the "action" that is recognized. Distance might be straight Euclidean(square root of the sum of the squares of the difference vector) or you might scale the the elements of difference vector differently because the magnitude of the moments are different depending upon the order $(p + q)$.

With the matching method you can create a 3x3 "confusion" matrix where the row i column j is the percentage of videos of action i that are best matched by a video of action j. Perfect matching would yield a diagonal matrix. You will try this using both the central moments μ_{pq} and the scale invariant ones η_{pq}.

2.1 Create the confusion matrix for the 3 actions for both the central and scaled moments. See if scaling the dimensions of the difference vector can achieve better results.

Output: Code and the best confusion matrices you can achieve. Describe what if any change you made to the distance function for matching to achieve this result.

So you probably just got very good confusion matrices. That's because we cheated. If you look at the best match for a given input, it probably was the same person doing the same action. Hardly a tough test. This time you're going to remove the test subject from the "training data". For example, if we remove subject 1 (P1) from the data, we will test on all 9 of P1's sequences and get a confusion matrix for P1. We can then do the same for P2 and P3. You may now find that you need to play with the scaling more carefully to get decent results. Or, you may find it very hard to get decent results. This is a new part of the problem set ....

2.2 Create the confusion matrices for each of the subjects as described as well as the average confusion matrix. So there are 4 confusion matrices, one for each subject shwoing the recognition results when not including that subject in the target data, and one matrix which is the averge of those 3.

Output: Code and the best confusion matrices you can achieve. Describe what you had to do to get these.

3 Extra credit: Detection and Recognition

All of the above involved taking pre-segmented videos and classifying them. A more realistic use of MHIs is to actually do detection and recognition.

There are two videos TestLabel1.avi and TestLabel2.avi. Each has a single person entering the field of view and doing some actions. Every now and then they do one of the three trained actions above. Your goal is to build a detection and recognition system.

The method is simple: continually compute the MHI through the video and the associated moment vector. At every time step compare it to the moment vectors of the MHIs of the training data. If the closest match is "close enough" you have detected the action. To get a decent detection measure you'll need to do some form of local suppression: if you detect the action ending at frame 50, you should not also announce a detection at frame 52. They are the same detection.

In classification we talk about *Recall* and *Precision*. Recall (or sensitivity) is the percentage of real entities actually detected. So if there are 6 actual actions in the test sequence (two instances of each of the three actions) and your system detected 4 of them, your Recall is 67%. (The two missed actions are "misses" or "false negatives".) Precision is the percentage of your announced detections that are real. So if you system thinks it detected 5 actions of which 4 were real, your precision is 80%. (The one error there is a "false alarm" of "false positive".) There are various measures of "accuracy". When real detects are rare compared to the number of frames, one measure of accuracy is called the F1 score:

$$F_1 = \frac{R \cdot P}{R + P}$$

where R is recall and P is precision.

3.1 For each test sequence, adjust your system to try to find the actual actions. See if you can adjust your system (a) to detect all the actions (100% Recall) with perhaps some false alarms (incorrect detections), (b) detect only real actions (100% precision) - no false alarms but maybe some misses, (c) to achieve the best F_1 score.

You should report your results as follows. For each of the settings (a, b, c) and for each of the two test sequences, report the recall, the precision and the F_1 score. So, for example, for (a) settings your recall should be 100% but the precision and F_1 will be whatever they are. Also for each setting and for each sequence report the detections found (both real and false). That is just list the frame numbers that result in a detection of each action.

Output: The code and (1) a listing of the statistics and detections in each of the cases, and (2) a description of what you had to do to make the thing work at all.