Lab 4: Motion Compensation and CBIR

Introduction

In this lab, you will attempt to implement two alternative template matching algorithms. The first is the Motion Vector Computation considered in the Motion Compensation algorithm within a video codec. The second is an image matching exercise, where you will be expected to index a small collection of images using colour histograms, then match a query image against this collection and order the collection in terms of similarity with the query.

Motion Vector Computation

In the event that frames do differ within a video sequence, one major cause of this is due to motion. Motion can be due to camera movement, or movement of objects in the scene. In either case, motion can cause large inter-frame differences, which translate into higher entropy for the P-frames. To reduce such differences, rather than use a fixed predictor location from the previous frame (as considered in Lab 4), one might make the assumption that pixels have undergone some motion. As such it is possible that another pixel in the neighbouring region of the co-located pixel might be a better representative for the predictor than the pixels immediately adjacent.

A second observation is that objects moving are probably larger than one pixel in size, thus groups of pixels tend to move together. As such, blocks are predicted from other nearby blocks in the previous frame. In general, the process of motion compensation is, given a target block position in the current frame, to then search within a small region (radius k) around the co-located block position of the previous frame (Fig. 1).

For a given target block, a set of candidate blocks $C_n[p+i, q+j]$ are tested in the search region. For each candidate, a metric is calculated assessing the 'similarity' or error between the candidate and the target. The Mean Absolute Difference (MAD) is one such metric:

$$MAD(i,j,p,q) = \frac{\sum_{p=1}^{m}\sum_{q=1}^{n}|C_{n+1}[p,q] - C_n[p+i,q+j]|}{mn} \tag{1}$$

In defining a MAD value for every candidate block location (i, j) in the search region, the actual motion vector is chosen as the (dx, dy) corresponding to (i^*, j^*) with the smallest MAD from the search region:

$$mv(p,q) = (dx, dy) = (\mathrm{i}^*,\mathrm{j}^*) \tag{2}$$

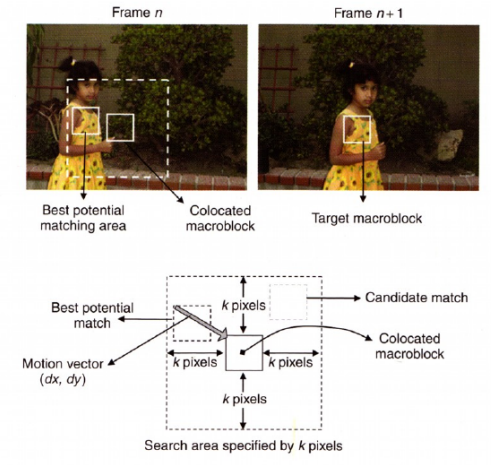

Figure 1: Motion Vector Computation

Content-Based Image Retrieval (CBIR)

Basic content-based image retrieval can be achieved as a template matching exercise, in which a database of images, and a given query image are each indexed using some form of descriptor.

We have considered colour histograms for this purpose in the lectures. Such descriptors essentially involve the calculation of a histogram on each colour channel (given a particular colour space is used to represent the image: e.g. RGB, YUV, YCbCr or HSV). A histogram typically divides up the intensity range into several levels (bins): for instance, the R channel from an RGB typically ranges over intensities 0-255. If we desire 20 levels, then we would first quantize the R channel into 20 uniform levels (as in Lab 1). A histogram can be calculated from the quantized R channel. Finally, the complete colour histogram is constructed by concatenating (joining) the histogram from each channel, together into a single 1D vector.

Matching between images can now be conducted using one of the following similarity (distance) metrics:

Manhattan (cityblock) distance:

$$d_{L1}(h,g) = \sum_i |h(i) - g(i)| \tag{3}$$

Euclidean distance:

$$d_{L2}(h,g) = \sum_i (h(i) - g(i))^2 \tag{4}$$

Histogram Intersection:

$$d_{int}(h,g) = \frac{\sum_i \min(h(i), g(i))}{\min(|h|, |g|)} \tag{5}$$

Laboratory Task

Part I: Motion Vector Computation

1. Collect or open a video file.
2. Extract the first and second frames from your video sequence (convert to grayscale). Pick two frames that have some (visible) motion between them.
3. Divide the second frame into a series of 16X16 blocks.
4. For each block in the second frame, perform a sequential search in the co-located region of the first frame with a radius of k=7. It is best to write a function that will compute a single given motion vector:

$$[dx, dy] = computeMotionVec(framePrev, frameCurr, p, q, k)$$

where (p, q) is the co-located block position, and k is the search radius. (dx, dy) will hold the values of the offset (in terms of pixel position) of the best matching block from (p, q).

5. Apply your motion vector computation to all blocks from the second frame (frameCurr). Calculate the DPCM using your motion vectors as predictors, compare the entropy against using a direct frame difference. Explain your result.

Part II: Content Based Retrieval

1. Collect at least 10-20 images (these may be from your own photo collection, or from google image search. Ensure that there are some images that share similarities in terms of colours, and some that appear (in your opinion), quite different in terms of colour.
2. Calculate histograms for your images using the HSV color space, 30 bins for H channel and 15 bins per channel for S and V. You can use built-in MATLAB/OpenCV functions to calcualte the histograms, or you can write your own function.
3. Using a single image from your database as a query, perform separate matching operations (using each of the three metrics in equations 3-5). Show the order in which images are deemed similar. Include a table with the distance measure results and a figure of the ordered set of images in your report. Justify the difference in results using the different similarity metrics.
4. Modify your code to divide each image into 8 equal blocks. Calculate Histogram individiually for each block, concatanate all the block histograms for a single image, and perform separate matching operations between images as in Step 3. Include the results in the same table (or a separate table) in step 3. Are you observing better results when using local histograms? Explain why/why not.

Submission: You must demo your lab to your TA during the lab session. All group members MUST be present. Submit report (PDF in IEEE format as per D2L), and source files on your D2L submission folder.

DEMO DUE: **Week of April 8 (last week of classes)**

REPORT DUE: **April 14 - 11.59 PM**

RUBRIC		5 (pts)	5 (pts)
		Code and Demo	Report