Intra- and Inter-frame Coding

ELE 725 Lab 3

Dylan Pereira
500512067
Ryerson University
Toronto, Canada
dylan.pereira@ryerson.ca

Nishantan Jegarajah
500519391
Ryerson University
Toronto, Canada
king.rajah.18@gmail.com

Abstract— **This lab report presents our findings on principles of two template matching algorithms, Motion Vector Computation of motion frames in videos and Content-Based Image Retrieval for images.**
The experiment was conducted in MATLAB software, where scripts are read into MATLAB command window to read a video file as input source and among two consecutive frames, search for the best pixel block candidate among a region and calculate the motion vector. To do so, the Mean Average Difference was calculated for multiple regions around a target region, once the best candidate was found, that candidate offset then becomes the motion vector for the block. For CBIR, scripts are read into MATLAB command window to generate images by loading collection of image with similar low and high frequency as input sources and matching a chosen query image against this collection in terms of similarity using colour histogram. Matching between images with respect to the query can be determined using similarity (distance) metrics provided: Manhatten distance, Euclidean distance, and Histogram intersection.

Keywords- MATLAB, Motion vector computation, Mean absolute difference, template matching algorithm, CBIR, content-base retrieval, matching descriptors distance metrics, Manhatten, Euclidian, Histogram intersection

I. INTRODUCTION/THEORY

In this lab the objective was to study and implement the two template matching algorithms, Motion Vector Computation and Content-Based Image Retrieval. In the Motion Vector Computation, we took the same video file that was used in LAB-3and extracted two grayscale frames. Both frames are then divided into 16x16 blocks, and given a search radius of 7, every block in the second frame is compared with the block in the first frame offset by the search radius going from -7 to 7. For each offset the MAD value was calculated and the minimum MAD value of all the offset will then determine the best matching block for the second frame. This process was done for all blocks of the image and hence used all the blocks best offset to generate a more accurate motion as compared to taking the difference of the two frames directly. It can be seen that the difference frame essentially plots the area in which there was a change in motion and hence can be utilized to maximize the compression rate for an video file.

For content based image retrieval (CBIR), the goal was to observe and understand the similarities of the content within several images with different colour spaces. CBIR searches through a database of images and indexes the best match in terms of similarity using colour histograms of all the images with respect to an image in the database chosen as the query or reference. Using the various different colour spaces: RGB, HUV, YcBcR, or HSV, the images two to three colour channels is quantized by a predefined number of levels for each channel in the colour space and concatenated together to create a 1D or 3D vector.

Matching between images was conducted using the following three similarities metrics: Manhatten (city block) distance, Euclidean distance, and Histogram Intersection. The collection of the chosen image database, includes some images that share common and some that are distinct, in terms of colour, this can better approximate the order of the intensity of colours between images. The RGB and HSV colour space histogram was used for matching the images in order with respect to the Query image based on the different similarity metrics. The results showed that HSV histogram generates better indexing of the image database in comparison to the RGB histogram, thus meaning that the RGB colour space can't be able to discriminate the level of the color intensity as well as the HSV colour space which contains more color intensity properties.

II. THEORY

A. Motion Vector Computation

To extend the ideas of image compression to video, temporal redundancy can be exploited. Often in videos, a sequence of images will contain a common set of visuals for several frames. Most components of the frame remain static in the background while those objects in focus in the foreground are dynamic. In some cases the entire set of data in a video frame will move due to the camera movement; pan, tilt or rotate. The changed parts of the image can be coded such that a motion vector can used to describe the transition of an object in the frame N to frame (N+1). As long as the object movements are not extremely fast a good approximation for a motion vector can be detected if:

$$C_{n+1}(x,y) \neq C_n(x,y) \qquad \text{or}$$

$$C_{n+1}(x,y) = C_n(x',y') = C_n(x-dx, y-dy)$$

Where x' and y' are in the local neighborhood of n.

By sweeping a local region around the collocated region of the previous frame, a match for the object can be found by choosing the most similar candidate to the current frame which is used to create a motion vector (dx, dy). The current frame, called the "target frame" will then be predicted from the previous frame called the "reference frame" with some added error term. Usually this is carrier out at a block level since temporal coherency is traceable for large clusters of pixels rather than individual pixels. The candidate selection process is performed using an algorithm called Mean Absolute Difference or (MAD) defined by the equation:

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

Where $C_n[\,p+i, q+j\,]$ are the candidate blocks of the reference frame collocated from the original point (p,q) by the value (i,j). Using the equation a MAD value is assigned to each candidate and the lowest value is chosen for the best match. The candidate scan region is defined by the radius from the point (p,q) and is user defined. Larger scan regions might provide more prediction range but that comes at a cost of more computation.

B. *Content-Based Image Retrieval*

Content-Based Image Retrieval (CBIR), also known as query by image content (QBIC), consists of retrieving the most visually similar images to a given query image from a database of images. CBIR is the process by which one searches for similar images according to the content of the query image, they are not searched based on keywords or annotations, but based on properties such as colour intensity, texture, shape, etc. When a specific descriptor is assigned the process will then narrow down and display the number of images in a database that is the best matches.

The content based image retrieval can be accomplished as a template matching algorithm. Colour histograms calculations were presented based on HSV color space. A typical RGB histogram divides the red, green and blue channel intensity over a range from 0 to 255. The HSV histogram will provide more color info than RGB histogram, that is HSV is a good colour space when trying to differentiate between two colours with different intensity. The three descriptors method that can match between images are shown below. Each method has different approaches to calculating the overall histogram, and hence will generate a different set of indexing order of database images; they can be interpreted as looking for different properties of an image like colour, shape and texture.

Ryerson Engineering. *(sponsors)*

Similarity Distance Metric Equations:

Manhatten (city Block) distance: (3)

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

Euclidean distance equation (4):

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

Histogram Intersection equation (5):

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

III. Methodology

This experiment was conducted by completely through MATLAB simulations. In the first part of the experiment, we create a function to perform motion vector computation on a video source. First we extract the one frame and its successive second frame of the video and display them. We then break the two frame into 16x16 pixel blocks and send them as parameters to call another function named blockMotion(). Inside blockMotion(), it will keep the second frame as constant and will offset the first frame image region by a radius: r=7 , looping through the offset ranging from -7 to +7. Doing this will allow us to generate the MAD value for each radius region. Once all the MAD values for each offset are calculated we then take the minimum MAD value from the matrix, and the location index of that minimum MAD value will predict the best matching motion of a single block with respect to its successive frame. This process was then repeated to search through every 16x16 block of the second frame and hence a motion vector computation was prediction for the entire region of the image.

In the Content-Based Image Retrieval, a database of images with varying colour intensity was used as input sources.

We first took an image and represent it in two colour spaces, RGB and HSV. We then break down the image into three different colour channels; R-Channel, G-Channel, B-Channel of RGB colour space and H-Channel, S-Channel, V-Channel of HSV colour space. The respective colour channels for both colour spaces were then displayed along with its normalized histograms. The next step we created a function that implements the three distance metric equations discussed earlier to calculate the similarity between two histograms. The function, calcSimilarity(), takes query image and the database images as parameters and calculates the similarity metric between the two images, we then arrange the similarity metric between the query image and database in ascending order of the best similarity. We used the function calcSimilarity() to find the best matching for a given query image that exist within the database and used the RGB colour space histograms first to generate the ascending indexing and then repeated the same steps and used the HSV colour space histograms for best matching with more levels(bins). The same steps was then repeated in terms of using the HSV colour space but this time using a different query image that didn't exist in the database and indexed the best matching colour histogram.

IV. Results

Figure 1 shows the first frame and figure 2 shows the second frame. Figure 3 shows the MVC difference frame and Figure 4 shows the direct difference frame, as you can see Figure better predicts motion but with high error due to blocks of images rather than pixels. Histogram testing is conducted in figure 5 and figure 6 for RGB and in figure 7 for HSV colour spaces respectively. The Query image of the turtle in figure 8 is used for RGB distance measurement for the first part of content based retrieval. The original data set that is unsorted is shown in figure 9. Figure 10, figure 11 and figure 12 show the Manhattan, Euclidean and Histogram Intersection distance measurement matches sorted by the most similar to the least similar. Figure 13, Figure 14 and Figure 15 show the same effect but with the HSV colour space. In Figure 16, Figure 17 and Figure 18, query image 2 'Lena.jpg' is used to match with HSV Manhattan, Euclidean and Histogram Intersection respectively for the last component of the experiment.

V. Discussion

A. *Motion Vector Computation*

The results of the best motion vector computation between figure 1 and figure 2 is shown in figure 3. As you can see when there is motion between frames of a video, it causes a discrepancy between corresponding pixels between frames. Even if there is a slight motion between frames the difference in terms of pixel values may be large. The DPCM between frames will be large anytime there is motion in a video. This large discrepancy makes encoding very inefficient due to the fact that there is a new pixel value for every slight motion. This makes compression ineffective when there is motion in every region of the video. In order to improve the compression rate when there is motion in a video we try and look for the best candidate for the co- located region in the following frame. This means that instead of looking at how much a certain pixel has changed when it transitioned from one frame to another we search for the best possible match to the pixel in the following frame. This pixel is the pixel that most likely moved to its new position when the frame changed. The distance between these pixels along with its direction of travel is calculated as a motion vector. To make compression even more effective we used 16x16 blocks of pixels instead of just individual pixels; however the downside of this is that the approximation error between frames will be higher, as can be seen from Figure 3 using MVC and Figure 4 using direct frame difference.

B. *CBIR*

In figure 5 we take the RGB histogram of the input image "water5.jpg" which is an undersea photograph. In the original image we primarily see different shades of blue and white and not much of any other colour. The histogram in figure 5 shows the content contains a large amount of blue channel data but also a large amount of green channel data. This is because the tone of blue that is seen in the water is a strong blend of the B and G channels. The R channel is mostly absent in this image since it appears very dark. The histogram shows how the R channel has very low contribution since the lowest quantized level makes up a significantly higher part of the histogram. The G channel histogram is more centered where as the B channel histogram is shifted furthest to the right proving that it is the dominant contributor. A combined histogram of all channels in figure 6 shows the net intensity levels of the image and gives information on the total image contrast. Here we see that a large portion of the image content is found in the right most end of high intensity values, but there is also a large contribution of extremely low intensity values. The mid region is not emphasized and the image is relatively high contest.

We then HSV transform the image to get the Hue, Saturation and Value information of the image. The hue image and histogram numerically identifies the location on the HSV cylinder that contributes to the colour makeup of the image. In this case, figure 7 the blue-green region is the result. The saturation channel shows the intensity of the colour away from a faded washed out look. There is a large spike in the saturation histogram at the high end which shows that the majority of the colour present is extremely saturated and pure. The value Image shows the relative brightness of the colours present thus it appears as a greyscale version of the image. The histogram of the value channel is similar to a blue channel histogram from figure 5.

In the content retrieval testing proved to successfully choose the correct matching image for the query image in Figure 8. We attempted to match this to a set of images including the original image that had similar visual looks. The original unordered image set from Figure 9 is sorted to that in Figure 10 when using the Manhattan Cityblock based distance measurement. The query image chose itself as the best match from the database which is trivial but is a first step towards asserting the functionality of the algorithm. When switching to the Euclidean distance based sorting the query image was once again chosen as best match as seen in Figure 11. However, there were some minor variances for the order from both techniques in the middle. The 3 least similar images were the same for both approaches. Euclidean distance is the absolute displacement between the two points of data and is slightly more accurate in measuring true multidimensional differences compared to Manhattan.

The result of Histogram Intersection based matching in figure 12 showed a result that was very different from than that in Euclidean and Manhattan based distance estimation. Histogram intersection looks at the parts of the histogram from the query and the sample that overlap to produce similarity metric. This method should work similarly to the Euclidean and Manhattan but the results were drastically skewed. If candidate images have histograms that are both very similar to the query, there can be problems detecting the best match.

Switching over to the HSV colour space, the results of Manhattan Cityblock matching are shown in figure 13, Euclidean in Figure 14 and Histogram Intersection in Figure 15. The order compared to the RGB based matching has changed slightly. The worst performing algorithm remains to be Histogram Intersection in Figure 15. In terms of most successful, the HSV space was better performing than the RGB space when the image colour tone was considered. This is likely due to the prominent contribution of the matching of saturation values. More specifically, Euclidean based HSV matching showed the best colour sorting ability in figure 14. We then replace the query image to something completely different and the skewed result for the Histogram Intersection remain as seen in figure 18. It seems as if the results are in reverse order; since the output seems consistently poor, the implementation is likely flawed. Note that in figures 16, 17 and 18 the query image is placed at the end as reference and is not part of any calculation. The result in HSV based Euclidian distance measurement continues to appear as the most accurate match in terms of visual colour pallet similarity.

C. Figures and Tables

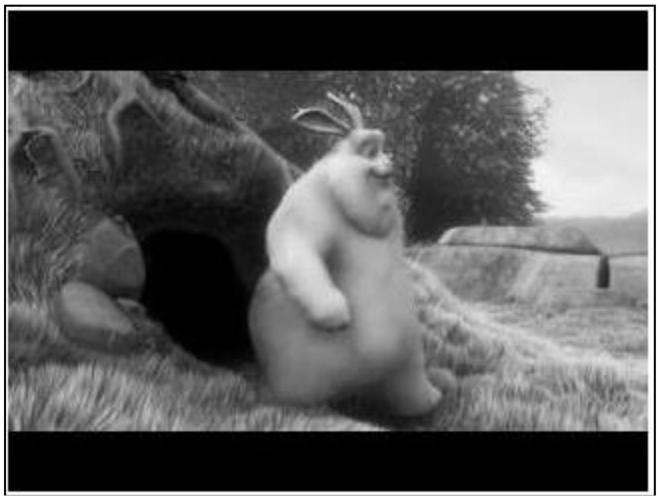
Figure 1 First frame of video sample 'bunny_gray.avi''

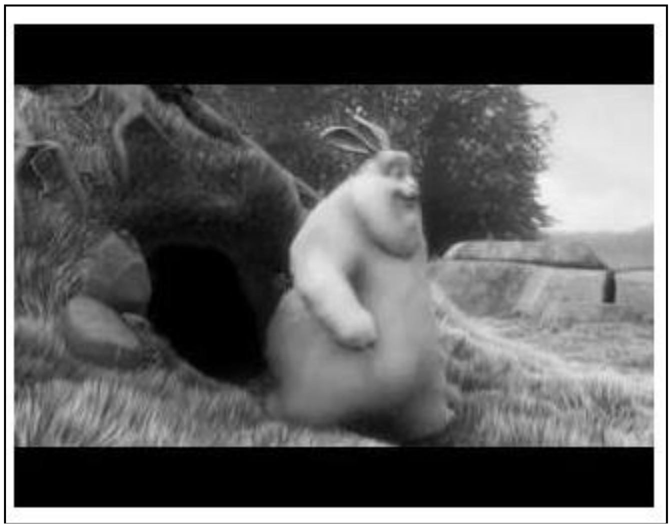
Figure 2 Second frame of video sample 'bunny_gray.avi

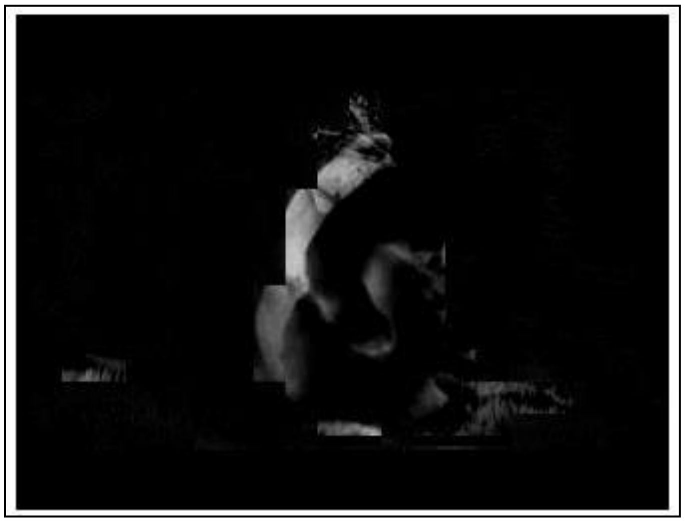
Figure 3 Motion Vector Computation Difference frame (between first and second)

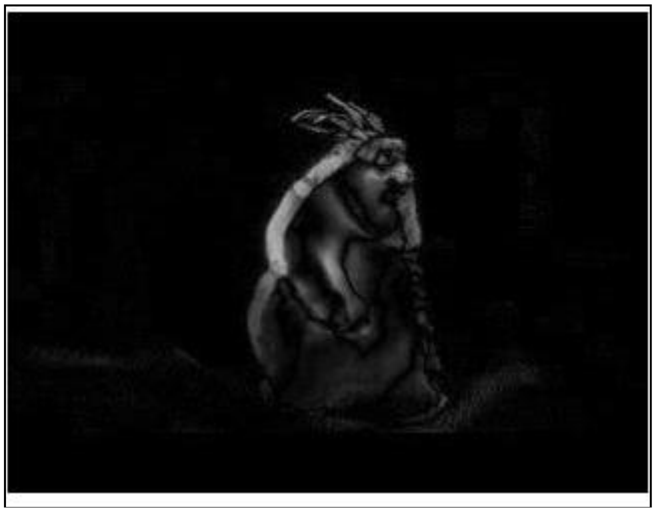
Figure 4 Direct Difference frame (between first and second)

Entropy	Entropy
Motion VC Frame Diff.	1.8081
Direct Frame Difference	3.1039

Table1: Entropy comparison

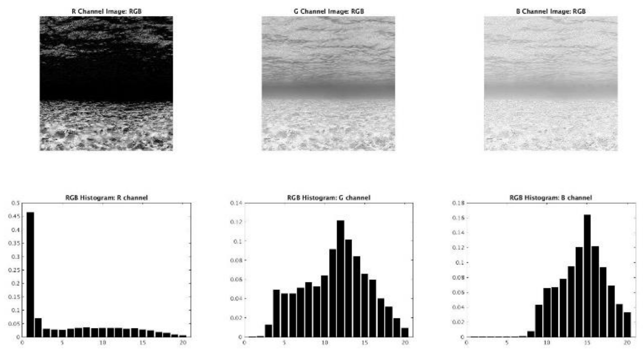

Figure 5 RGB Channel Histograms

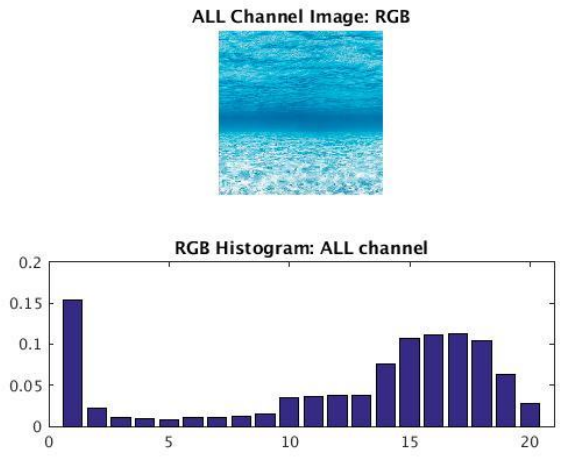

Figure 6 RGB combined Histogram

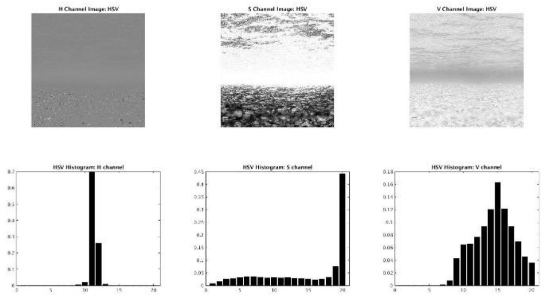

Figure 7 HSV histogram

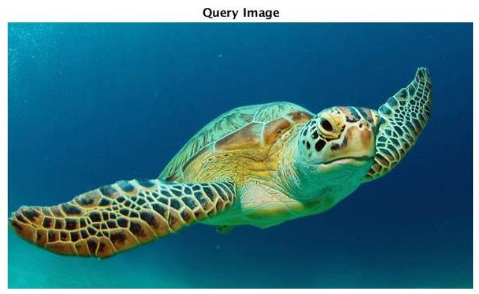

Figure 8 Query Image 1

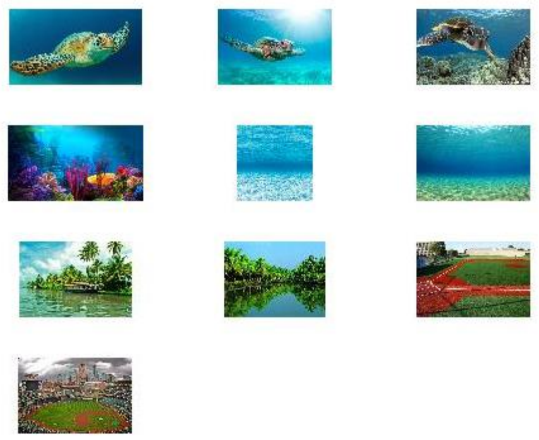

Figure 9 Unsorted Database

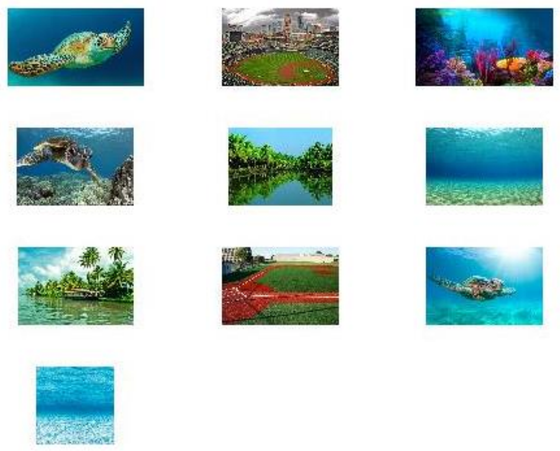

Figure 10 Manhattan RGB

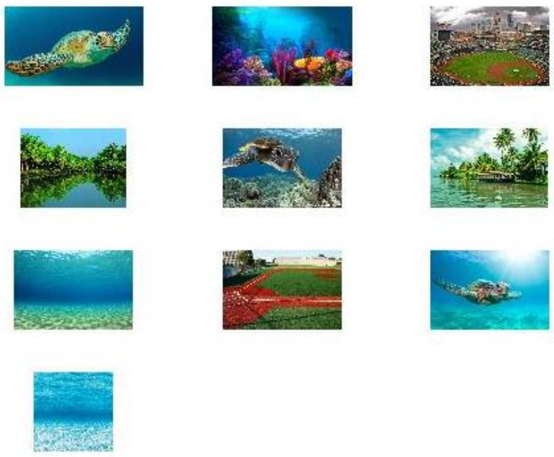
Figure 11 Euclidean RGB

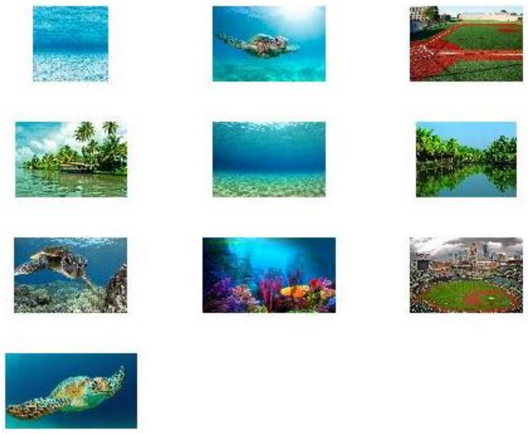
Figure 12 Histogram Intersection RGB

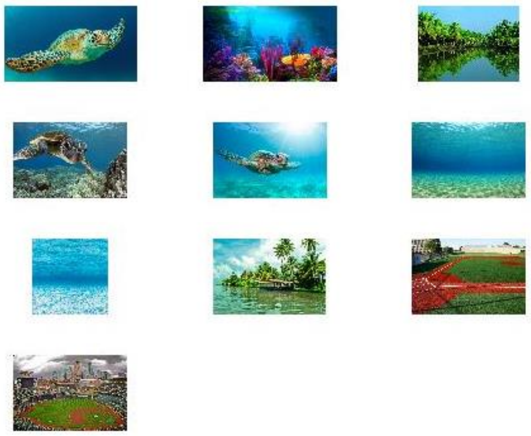
Figure 13 Manhattan HSV

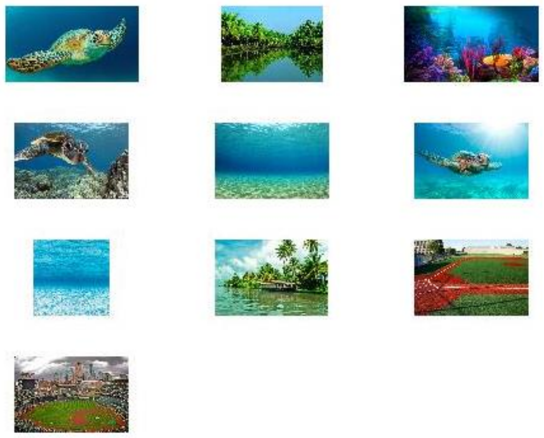
Figure 14 Euclidean HSV

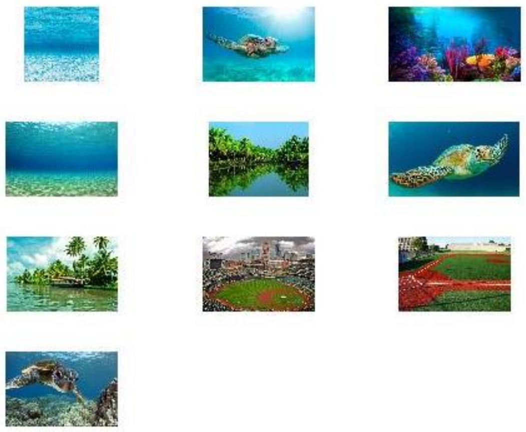
Figure 15 Histogram Intersection HSV

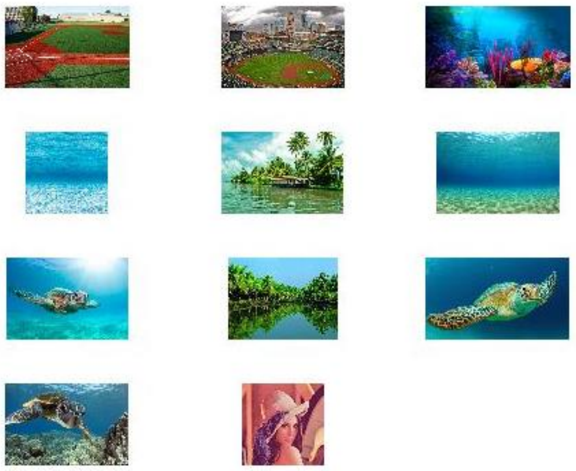
Figure 16 Query 2 Manhattan HSV

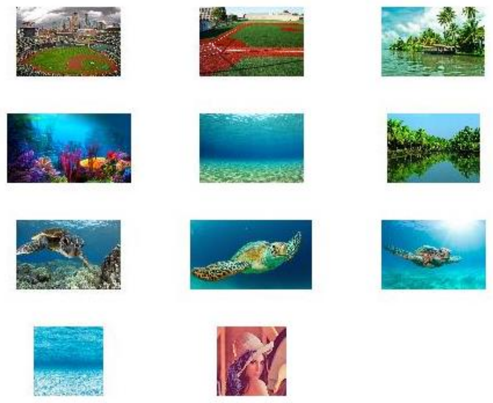
Figure 17 Query 2 Euclidean HSV

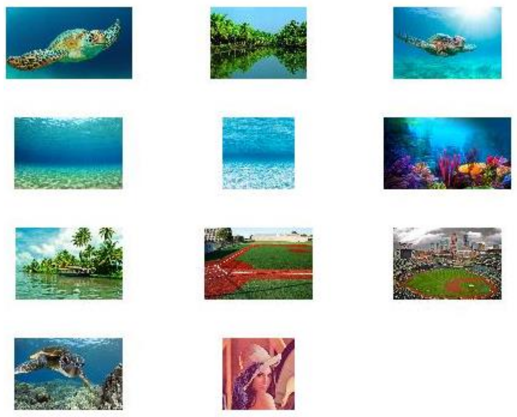
Figure 18 Query 2 Histogram Intersection HSV

VI. CONCLUSION

In this experiment we were successfully able to implement two template matching algorithms. We managed to extract two grayscale frames with visible motion from a video. We were able to calculate the MAD for each candidate block given a specific radius within the image border. From this data we calculated the motion vector between these two pixel positions and use that as an offset for generating the difference frame.

We were able to successfully use a method of comparing a query image to a set of images and comparing the similarities between them and order according to the best match. As expected, the HSV color space provides more accurate similarity matching than the RGB for all the three similarity distance metrics.

REFERENCES

[1] Khan, N. (n.d.). Basics of Multimedia Systems. Retrieved March 23, 2017.ELE725_L07.1_VideoCompressionELE725_L04.1_quantizationandaudio Lecture Slides

[2] Khan, N. (n.d.). Basics of Multimedia Systems. Retrieved March 23, 2017. ELE725_L07.2_VideoCompression Lecture Slides

[3] Khan, N. (n.d.). ELE725 Lab Manual. Retrieved March 23, 2017.