A NEW ENERGY FUNCTION FOR SEGMENTATION AND COMPRESSION

Michael King, Zinovi Tauber, and Ze-Nian Li

School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, CANADA

ABSTRACT

We propose a new DCT-based energy function E_{DCT} for object-oriented segmentation and compression. By minimizing E_{DCT} the best possible split of the image blocks can be found which leads to better segmentation and compression. Consistent motion vectors can be obtained by using an extended 3D energy function, in the spatio-temporal domain, which measures block motion over multiple frames. Tests on images and videos show promising results, where still image compression achieves over 15% improvement over JPEG and video compression achieves similar improvement over MPEG-2.

1. INTRODUCTION

Object-based video encoding methods take objects, ideally real world objects, and encode them independently [1]. These methods have the potential to improve the compression rate and be more scalable. However, it is known that automatic *object-based* coding is difficult due to the hardship of extracting objects. Digital video encoding uses the *Discrete Cosine Transform* (DCT)[2]. We propose a DCT-based energy function that groups pixels into regions of similar textures. These regions facilitate a more efficient image and video compression than global methods.

Generally, blocks within a region move in similar directions in a video. This property can be used to achieve higher compression due to increased accuracy in temporal prediction. Since each region is encoded independently, object in the scene are scalable; Details can be sacrificed from unimportant objects in the scene, such as the background, while maintaining high visual quality for important objects, such as the main actors in a movie. Furthermore, interactivity can be extended to allow the viewer of the digital video to manipulate the appearance and movements of the video objects.

2. NEW ENERGY FUNCTION FOR SEGMENTATION

Unlike traditional image encoding which consists only of square blocks of 8×8 or 16×16, object-oriented image encoding allows *arbitrarily shaped blocks* (ASBs). The central assumption for DCT-based image compression is that natural objects contain mostly low frequencies, and so the high frequencies can be efficiently encoded using 0-runs [3]. Square blocks can be split into ASBs of similar texture patterns by minimizing combined affinity for high frequencies. These ASBs should also correspond to separate objects. Once the ASBs are created, a pyramidal linking scheme can be used to grow the regions.

A DCT-based energy function is created and defined as:

$$E_{DCT} = \sum_{i=1}^{w} \sum_{j=1}^{h} e^{[(\frac{ij}{wh})^2 - 1]} |DCT(i-1, j-1)|, \quad (1)$$

where E_{DCT} is the energy of the block, w and h are the width and height of the block, and $DCT(i, j)$ is the (i, j)th DCT component when $i + j > 2$, and 0 otherwise.

The energy function maps the texture from a multiple-dimensional frequency space into a one-dimensional energy space. It assigns exponentially higher costs to higher DCT frequencies, since we expect the highest frequencies to be caused by a mixture of objects. The DC value is treated separately since it is color dependant and does not influence the texture. The energy function is used to determine the likelihood of pixels in a block belonging to a single region, hence the average color does not influence it and is not considered here.

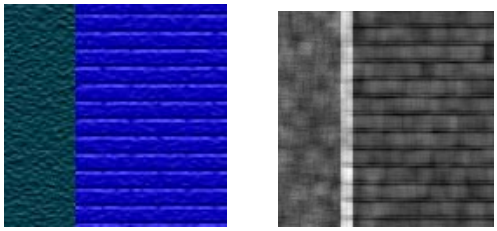
A highly textured block of pixels produces a high energy value from Eq. 1, due to having many high frequency values; Conversely, a low textured block of pixels, like a block of homogeneously colored pixels, produces very few high frequency DCT values and results in a low energy value. Even an ordered artificial texture block would produce a lower energy value than an unordered one, by generating fewer high frequencies. Thus, we find the energy function to be broad enough to support most man-made, synthetic, and natural scenes.

An important feature of DCT to note is that if the group of pixels being compared is comprised of two distinct textures, it will produce many high frequency values. For example, if the pixels being processed are composed of a black group of pixels and a white group of pixels, the edge between the regions will need many cosine waves to represent it, resulting in numerous high frequency DCT components. Similarly, the energy value of an edge between any two distinct textures is expected to be significantly higher than either texture processed separately as shown in Fig. 1. An algorithm to find the division between textures in an image can take advantage of this property by comparing energy values.

Finding the best block subdivision is a minimization problem over E_{DCT}. Stated informally, we are looking for any number of ASBs for which the combined E_{DCT} measure is the lowest over all configurations of the block splits. The minimization can be stated as:

$$E_{DCTtotal}(b) = min(\textstyle\sum_{r \in R} E_{DCT}(r)),$$
$$\text{for all possible sets of ASBs } R \in R_T, \quad (2)$$

where R is a partition of a square block into ASBs and R_T is the set of all possible partitions of a square block. This minimization is computationally expensive, so we apply a series of approximations to obtain near optimal initial conditions.



(a) Two texture regions (b) Feature energy space

Fig. 1. A 256×256 pixel image. The gray level indicates the relative magnitude of energy in (b). Energy is low in homogeneous texture regions, while high between them.

Image segmentation using the energy function can be broken into two major steps: the first step is to create ASBs by finding region borders; the second step is to join these ASBs into regions. The boundaries of the ASBs are calculated using the energy function (Eq. 1) which characterizes the complexity of the texture. The energy function with a block size of 8×8 is calculated for every pixel in the image. The energy value is then assigned to the center pixel and a DCT energy map is created. Edges are detected in the energy map which vote for block division boundaries. Once the divisions are found and refined, the ASBs generated are used as the building blocks for a region growing algorithm [4].

3. EXTENSION TO 3D ENERGY FUNCTION

3.1. 3D DCT Motion Energy

Many motions in videos can be approximated by finding a linear motion which best fits temporally nearby frames. As with images, it is more desirable to work in the frequency domain than the spatio-temporal. Slow and small changes in objects are more likely than large ones. Instead of using mean squared error (MSE) as the measure for the accuracy, a novel motion detection based on 3D DCT is used, using an extension of Eq. 1,

$$E_{3DCT}(MV) = \sum_{i=1}^{w}\sum_{j=1}^{h}\sum_{k=1}^{t} e^{[\frac{i^2 j^2 k^2}{w^2 h^2 t^2} - 1]} \times$$
$$DCT(i + MV_x(k) - 1, j + MV_y(k) - 1, k), \qquad (3)$$

where w is the width, h is the height, t is the number of frames, MV is the single motion vector from frame 1 to t, $MV_x(k)$ and $MV_y(k)$ is the motion vector at frame k in the horizontal and vertical direction, respectively, and $DCT(i, j, k)$ is the 3D DCT.

A low E_{3DCT} energy indicates that either the motion vector predicted is slightly inaccurate or pixel values changed gradually due to shading. Slow changes, like changing illumination, would have very little effect on the 3D energy if it was approximately equal across the block over time. This energy function is well suited for gradient descent search by virtue of phase shifts being encoded efficiently by DCT. DCT coefficients in one dimension of shifting texture have a low frequency pattern in the second dimension.

Motion estimation has two major phases. First, the optimal motion for each individual block must be estimated, and second the optimal motion for each region must be recovered from the block motions estimated.

3.2. Motion Detection

We used Diamond Search to find a motion vector for each block. We set $t = 4$ so the motion search has 1/4 pel precision. The energy in Eq. 3 is calculated for each motion vector for the Diamond Search. In many cases the assumption holds true that the energy decreases as it reaches the global minimum.

Naturally, when objects move in a video all their parts move in similar directions. To find the best motion vector for a block in a region, all blocks in the region must be considered. This can be formalized as finding the MAP hypothesis in a Bayesian Framework with Gibbs distribution which is equivalent to minimizing the following,

$$E_{motion}(M) = E_{data}(M) + E_{smooth}(M), \qquad (4)$$

where E_{data} is the data term given by the energy of motion vectors M and the blocks, and E_{smooth} is the smoothness prior given by the discrepancy between neighboring motion vectors.

$$E_{data}(M) = \sum_{b \in B} E_{3DCT}(M(b)), \qquad (5)$$

where B is the set of all ASBs in the region, and

$$E_{smooth}(M) = \sum_{a \in B, b \in \mathcal{N}(a)} V(M(a), M(b)), \qquad (6)$$

where $V(M_a, M_b)$ is the weighted difference of the MVs of blocks, and $\mathcal{N}$ is a neighborhood relation.

We use the Graph Cuts algorithm [5] to estimate the best motion set for all the blocks in the region.

4. OBJECT-ORIENTED IMAGE ENCODING

The technique we use here essentially follows the digital image compression standard JPEG. The bit stream encoded is made up of information grouped by each region in the image. Each region has two components: the shape of the region, and the texture data of each block in the region. The shape is encoded into *Binary Alpha Blocks* (BABs) [7], while the texture is encoded using DCT. An overview of the steps involved in image encoding is shown in Fig. 2.

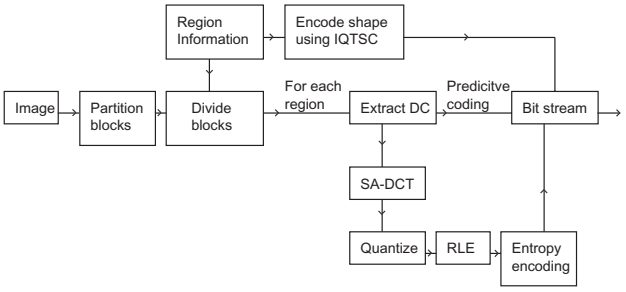
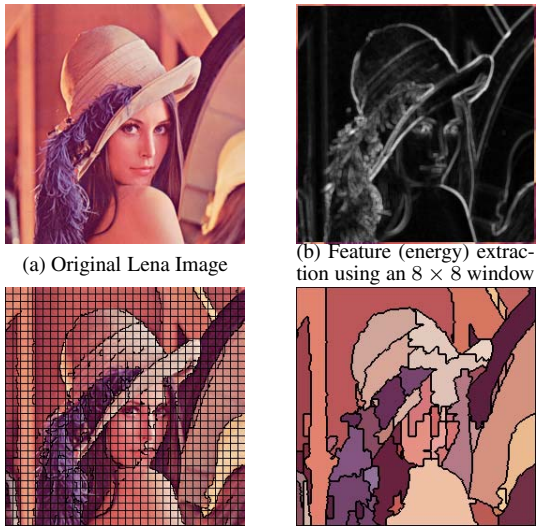


Fig. 2. Overview of the image encoding process.

All blocks are encoded using mean corrected SA-DCT [6]. Segmenting blocks into homogeneous textures and growing regions has the advantage of an increased compression rate. Blocks in a region would have similar DCs, thereby decreasing DC prediction errors to near 0. Additionally, block sizes can increase and represent the same texture without adding coefficients, reducing redundancy. Lastly, accurate block divisions allow multiple blocks encoded with SA-DCT to have higher compression than a single DCT encoded block.

For Shape Coding, we use Improved Quadtree-based Shape Coding (IQTSC) [7]. In IQTSC each node needs only one bit to represent its state. If the block size is greater than one pixel, it is encoded as either homogeneous or heterogeneous. If, however, the block consists of one pixel, then it is encoded as either opaque or transparent. An improvement was added to make the method lossless by adding a special bit code for the case that the predictive coding is incorrect.

The image encoding algorithm is computationally more expensive than JPEG, but scales well with size. It's computational complexity in relation to the area of an image is equivalent to other standard compression methods, which is $O(n)$.



(a) Original Lena Image (b) Feature (energy) extraction using an 8×8 window

(c) ASB division (d) Regions

Fig. 3. Lena Image and segmentation results.

5. OBJECT-ORIENTED VIDEO ENCODING

As in standard object compression methods, two types of frames must be encoded, I-frames and P-frames. The encoding of I-frames essentially follows the image encoding described in the previous section.

The bit stream encoding a P-frame is separated into regions just like an I-frame. For each existing region, the motion, shape, and error need to be encoded. For each new region, the information needed is the same as for the I-frame.

The motion is generated using a multi-resolution Graph Cuts method to increase the search area while using as few labels as possible. The image resolution is reduced by a factor of 4, and Graph Cuts is performed on a 9×9 search area, resulting in an effective 33×33 search area. Once the best label is found, Graph Cuts is applied in the 4×4 area around the best label in the full resolution image.

The motion vectors, except for the first one, are differentially predicted. They are predicted from their top left neighbors. Only pixels where an object expands or contracts need shape coding. The shape is encoded using IQTSC. The error of the frame is encoded as a regular image.

The average running time of the most computationally expensive operation, Graph Cuts, has been shown to be linear [5]. Therefore, the typical computational complexity of this encoding is equivalent to other standard video encoding methods which are $O(n)$ in the size of the area of the video.

6. EXPERIMENTAL RESULTS

One of the test images Lena (512×512) is shown in Fig. 3a. Segmentation of this image is challenging in many respects. The feathers in the hat and the hair have many edges and varying colors. The skin, hat and parts of the background have similar colors. There is also a strong directional light causing widely varying skin color, noticeably on the shoulder.

The energy map is shown in Fig. 3b in which each E_{DCT} is calculated for a block centered in the image. Gray scale is used to indicate the level of energy. Afterwards, good partition of ASBs and segmentation of regions are obtained (Fig. 3c and 3d).

Our object-oriented image coding was tested on other images, and in general it achieved over 15% improvement over JPEG. The compression results are shown in Table 1.

Table 1. Comparative image compression results in bits

Image Name	Bits for shape	Segmented Total	JPEG
Lena	19701	608017	739408
Parrot	15671	639365	742560
Baboon	20190	856022	1169846
Test Image 1	50	5337	8224
Test Image 2	602	3568	14543

The following presents analysis for the test video shown in Fig. 4. The MV energy for 3 blocks is shown in Fig. 5. Although the MVs might not always consistently decrease towards their global minimums, it is clear that even in this large motion, they are generally decreasing until they reach the minimum energy as shown in Fig. 6. Once the search is within a few pixels of the true motion, the MVs indeed descent faster towards the minimum energy. Fig. 7 shows the improvement over Diamond Search.

Fig. 4. Test videos of a fast moving toy car and a coast-guard ship.

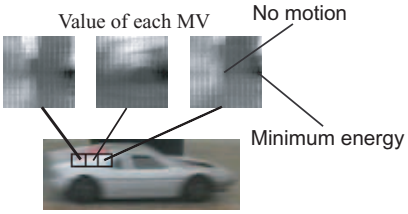


Fig. 5. 3D motion cost for three blocks. The higher the energy, the brighter the pixel on the 3D motion plane.

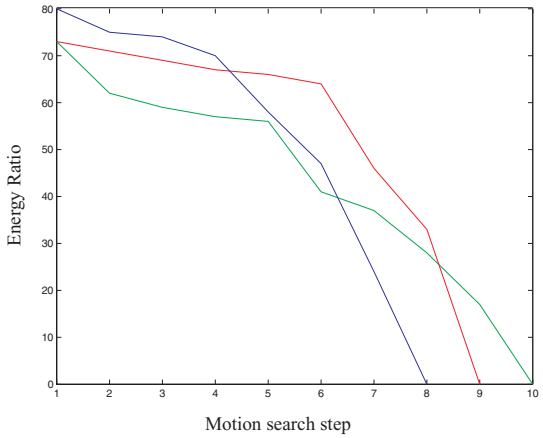


Fig. 6. This graph shows how the motion search finds the minimum energy in Fig. 5. Each jump in the Diamond Search lowers the Energy Ratio until it reaches the minimum energy. The Energy Ratio is defined as the current energy minus the minimum energy in the search window, then divided by the maximum energy of the search window.

Fig. 8 shows MV matching results from other test video clips. Table 2 shows some comparative results. To keep quality the same between encoding methods, a similar MSE was used while encoding using the object-oriented video compression and MPEG methods. Here, Diamond Search was used for MPEG-2 as well.

7. CONCLUSION

A novel DCT-based energy function is proposed for object-oriented segmentation and compression. The results are encouraging. Although true object segmentation is not achieved, the gap between image segmentation and object segmentation is being bridged. Improved compression is achieved by segmentation from compression principles.

Increased image compression may be achieved by increasing block sizes within a region, as well as by predicting block textures from neighbors within a region and encoding the error in a similar manner to motion compensation in standard video compression. Improvements to the video compression may also be achieved by integrating temporal information in the segmentation.

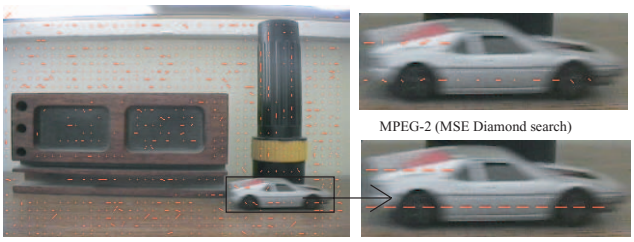


Fig. 7. The direction for each undivided block is shown with a red arrow. The best direction is found using diamond search which is unreliable in terms of true motion. When the energy is minimized for a region, good motion vectors are found.

Table 2. Comparative video compression results in bits

Video Name	PSNR	Segmented Mbps	MPEG-2 Mbps	Frames
Toy Car	44.9	0.324	0.389	142
Coast Guard	42.5	12.9	15	138
Synthetic video	53.3	0.004	0.033	150

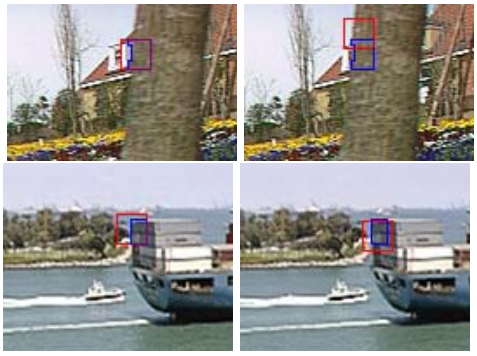


Fig. 8. More test video clips and the MVs between two frames. The red outline represents a MPEG-2 (MSE Diamond Search) tracked block, and the blue outline represents a better tracked ASB using the 3D energy function.

8. REFERENCES

[1] Fernando Pereira and Ebrahimi Touradj. *The MPEG-4 Book*. Prentice Hall, New Jersey, 2003.

[2] Al Bovik. *Handbook of Image & Video Processing*. Academic Press, San Diego, 2000.

[3] James F. Blinn. What's the deal with DCT? *IEEE Computer Graphics and Applications*, 13(4):78, 1993.

[4] Ze-Nian Li, Zinovi Tauber, and Mark S. Drew. Locale-based object search under illumination change using chromaticity voting and elastic correlation. In *Int. Conf. on Multimedia and Expo (ICME2000)*, volume 2, pp. 661–664, 2000.

[5] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[6] Paul Kauff and Klaas Schuur. Shape-adaptive DCT with block-based DC separation and DC correction. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(3):237–242, 1998.

[7] Mei-Juan Chen, et al. Multi-resolution shape coding algorithm for MPEG-4. *Int. Conf. on Consum. Electr.*, pp. 126–127, 2000.