# Hierarchical Minimum Spanning Trees for Lossy Image Set Compression

Anthony Schmieder, Barry Gergel, and Howard Cheng
Department of Mathematics and Computer Science
University of Lethbridge, Alberta, Canada

Xiaobo Li
Department of Computing Science
University of Alberta, Alberta, Canada

**Abstract**

Minimum spanning tree algorithms have been proposed for the lossy compression of image sets. In these algorithms, a complete graph is constructed from the entire image set and an average image, and a minimum spanning tree is used to determine which difference images to encode. In this paper, we propose a hierarchical minimum spanning tree algorithm in which the minimum spanning tree algorithm is first applied to clusters of similar images and then it is applied to the average images of the clusters. It is shown that the new algorithm outperforms the previous image set compression algorithms for image sets which are not very similar, especially at lower bitrates. Furthermore, the computational requirement for a minimum spanning tree is significantly lower than the previous minimum spanning tree algorithms.

## 1 Introduction

Traditional image compression algorithms for individual images, such as predictive coding and transform coding, have been shown to effectively reduce coding, inter-pixel, and psychovisual redundancy [5]. Image sets, however, may contain inter-image redundancy, or "set redundancy" [6], which are not reduced by these algorithms. Some work has been done to address this issue. The centroid [6], MST [1, 8], and MST$_a$ [3, 4] algorithms have been shown to reduce inter-image redundancy in sets of similar images.

In cases where images in a set form multiple clusters of similar images, there is potential for improvement. With the centroid and MST$_a$ algorithms, only one average image is calculated for the entire set of images. As the number of distinct clusters in a set increases, the average image becomes "less similar" to any single image in the set, and is therefore a less effective predictor for the

images in the set. This can negatively impact compression performance of the image set compression schemes.

The hierarchical approach presented here combines the $\text{MST}_a$ algorithm of Gergel *et al.* [4, 3] with the clustering algorithm of Nielsen *et al.* [9], by partitioning a set of images into clusters, and performing the $\text{MST}_a$ scheme on each of the clusters. The $\text{MST}_a$ scheme is then applied to the set containing the average images of each cluster.

Compression performance of the hierarchical algorithm is examined and compared with the compression performance of the traditional, centroid, MST, and $\text{MST}_a$ compression schemes. The run time of performing the clustering scheme on a set of images, and running $\text{MST}_a$ on the individual clusters versus the run time of the $\text{MST}_a$ scheme on the entire set is also analyzed.

## 2 Previous Work

Karadimitriou and Tyler proposed the centroid and min-max "set mapping" schemes to reduce "set redundancy" for lossless compression [6, 7]. The centroid scheme involves computing an average image for a set of similar images, calculating the difference between the average image and each image in the set, and coding the average image and the difference images. In the min-max scheme, a minimum image and a maximum image are created from the minimum and maximum pixel vales across all images. Several methods may be used to predict each original image from the minimum and maximum images. The minimum image, maximum image, and the prediction error for each image are coded. Their algorithm gave significant improvement in compression ratios compared to compressing individual images. However, the images in the set must be quite similar if the centroid and min-max algorithms are to perform well, and image sets that contain dissimilar images are not considered. To ensure similarity of the test images, clusters of ten images were selected from a larger set using a simple genetic algorithm, and each cluster is compressed independently. This algorithm runs quickly and produces clusters that are are quite similar. The clustering algorithm, although very effective for experimental purposes, is not practical in all environments. It requires the desired number of images in the cluster as input, which may be impractical to determine for large sets of images. Also, due to the random nature of the algorithm, there is no optimality guarantee for the output cluster.

Nielsen *et al.* proposed a clustering strategy that is adaptive to image sets containing dissimilar images [9]. In their approach, the root mean square error (RMSE) between images in the set and the average image is used to partition the set into clusters of similar images. Each cluster is compressed independently using the centroid scheme. JPEG2000 (lossless and lossy) [2] is used to compress the average and difference images. Their results are compared to "traditional" JPEG2000, which refers to using JPEG2000 to compress each image individually. The results of their experiments are encouraging, showing a 13% to 25% improvement over traditional JPEG2000. Compression performance is

clearly improved, but the effect of clustering on the run time performance of the algorithms was not examined. Also, their experiments did not compare the clustered centroid scheme with the centroid scheme on the entire image set, and did not consider other set mapping strategies.

The minimum spanning tree (MST) set mapping strategy, proposed by Nielsen and Li, is based on a graph data structure [8]. A complete graph is constructed, using images as the vertices and the RMSE between adjacent images as the edge weights. An MST for the graph is calculated, and one image is chosen as the root. The root image and difference images represented by the edges with the lowest total cost are encoded using lossy JPEG2000 [2]. The results of these experiments showed a clear improvement in average distortion (RMSE) when using the MST scheme over compressing each image individually, especially at lower bitrates. These experiments focused on sets of similar images, and did not examine performance on sets containing dissimilar images. They also did not compare the MST strategy with other set mapping strategies, such as the centroid scheme, and did not consider the effect of clustering on the MST results. Chen *et al.* [1] also applied a similar strategy to "object movies," which are sets of images of an object photographed at different pan and tilt angles. The prediction errors from motion compensation are used as the edge weights. They showed a clear improvement over previous methods. However, their algorithm is designed specifically for object movies where the image set is assumed to be similar.

Gergel *et al.* built upon this work with the $\text{MST}_a$ scheme [3, 4]. An MST is computed on a complete graph that includes a zero image and an average image, using RMSE as edge weight. The $\text{MST}_a$ scheme is a unified framework that adaptively chooses the best scheme among the traditional, centroid, and MST schemes. Gergel *et al.* compared lossy and lossless compression results between the traditional, centroid, MST, and $\text{MST}_a$ schemes. The $\text{MST}_a$ scheme is shown to be highly effective, outperforming the other schemes in many cases. For image sets which are very similar, the $\text{MST}_a$ strategy makes use of the average image to arrive at a strategy very close to the centroid strategy. On the other hand, for image sets which have clusters of similar images, the $\text{MST}_a$ strategy essentially chooses to compress each image independently because the average image for the entire set is not a good predictor of the images in the set. Furthermore, it may not be practical to construct the complete graph for a large image set.

# 3 Approach

The existing set mapping strategies have been shown to be effective on sets of similar images, but the images may not be similar in all cases. In the hierarchical $\text{MST}_a$ scheme, we partition the image set into clusters of similar images, and apply the $\text{MST}_a$ algorithm to each cluster. Since the average image for each cluster should be very similar to all images in the cluster, it is a good predictor for images in the cluster. This should produce difference images with a small

range of pixel values that will compress well. The goal is that the increased compression performance for the difference images offsets the added cost of storing multiple average images.

### 3.1 Graph Theory and MST$_a$

Gergel *et al.* described the MST$_a$ set mapping scheme as follows [3, 4]. Let $S = \{I_1, I_2, \ldots, I_n\}$ represent a set of $n$ images of identical dimensions. Two additional images are defined: the zero image $I_{n+1} = I_z$ where $I_z(i,j) = 0$ for all values of $(i,j)$, and the average image $I_{n+2} = I_a$ where

$$I_a(i,j) = \frac{1}{n}\sum_{k=1}^{n} I_k(i,j).$$

The average and zero images are added to $S$ to create a new set as follows:

$$S_a = S_{n+2} = S_n \cup \{I_z\} \cup \{I_a\}.$$

Gergel *et al.* defined a complete, undirected, weighted graph $G = (V, E)$ from $S_a$. The vertices of $G$ are $V = \{I_i | I_i \in S_a\}$. The edges are $E = \{(I_i, I_j) | I_i, I_j \in V\}$. The weight for each edge $(i,j)$ is defined as $w(I_i, I_j)$ where $w : S_a \times S_a \rightarrow \mathbb{R}_{\geq 0}$ is a function that measures the cost to reconstruct $I_j$ assuming $I_i$ is known. For this paper, RMSE is used as the weight function and it is symmetric.

The minimum spanning tree $T$ of $G$ is calculated, and the difference images represented by the edges in $T$ are coded.

### 3.2 MST$_a$ and Clustering: The Hierarchical MST$_a$

In this work, we add clustering to the MST$_a$ scheme to form a hierarchical MST$_a$ (HMST$_a$). The set of images $S$ is partitioned into $k$ clusters $S_1 \cup S_2 \cup \ldots \cup S_k = S$ where $S_i \cap S_j = \emptyset$ for $i \neq j$. Each cluster $S_i$ contains $n_i$ images, such that $S_i = \{I_{i,1}, I_{i,2}, \ldots, I_{i,n_i}\}$ and $n_1 + n_2 + \ldots + n_k = n$. The MST$_a$ algorithm is applied to each cluster. In the first step of the HMST$_a$ algorithm, two images are added to each cluster $S_i$ to form the set $S_{a,i}$:

- the zero image $I_{i,z}$ as $I_{i,z}(i,j) = 0$ for all $(i,j)$;
- the average image $I_{i,a}$ as $I_{i,a}(i,j) = \frac{1}{n_i}\sum_{l=1}^{n_i} I_{i,l}(i,j)$.

Next, a new cluster $S_A$ of the average images of all other clusters is created as $S_A = \{I_{1,a}, I_{2,a}, \ldots, I_{k,a}\}$ with $|S_A| = k$. The average image and zero image are added to $S_A$ to form $S_{a,A}$:

- the zero image $I_{A,z}$ as $I_{A,z}(i,j) = 0$ for all $(i,j)$;
- the average image $I_{A,a}$ as $I_{A,a}(i,j) = \frac{1}{k}\sum_{l=1}^{k} I_{l,a}(i,j)$.

An MST is computed from the complete graph constructed from each cluster $S_{a,i}$ as well as $S_{a,A}$. The difference images represented by the edges in the MSTs are coded. Notice that the resulting edges may not be a spanning tree for the complete graph constructed from the image set $\left(\cup_{i=1}^{k} S_{a,i}\right) \cup S_{a,A}$ because there may be cycles involving the average images $I_{i,a}$. These cycles are broken by removing edges connecting $I_{i,a}$ to obtain a spanning tree.

### 3.2.1 The Clustering Algorithm

For these experiments, we implemented the clustering algorithm described by Nielsen and Li [9]. Their algorithm sorts images into clusters based on both the percentage of pixels outside of the range $[-127, 127]$ in the differences between the images in the cluster and the average image for that cluster, and the RMSE between the images and the average image. Let $S_k$ represent the $k$th cluster. Let $I_{k,a}$ represent the average image for $S_k$. Let $\Delta(I_1, I_2)$ represent the RMSE between images $I_1$ and $I_2$, and $\%(I)$ represent the percentage of pixels in image $I$ that are in the range $[-127, 127]$. A percentage threshold, $\phi$, is chosen. For each image $I \in S_k$, if $\%(I_{k,a} - I) < \phi$, then $\Delta(I_{k,a}, I)$ is computed. The image $I$ in cluster $S_k$ with the highest $\Delta(I_{k,a}, I)$ is moved to cluster $S_{k+1}$. $I_{k,a}$ is then recalculated, and the comparison is repeated for images remaining in $S_k$. These steps are repeated until a pass is made through $S_k$ where no images are removed. This process is repeated for all clusters. The clustering algorithm described is presented in Algorithm 1 [9].

**Algorithm 1** Pseudocode for Nielsen and Li's clustering algorithm.

```
create S_0, containing all images
k ← 0
repeat
  repeat
    create average image, I_{k,a}
    max_rmse ← −1
    rm_img ← NULL
    for all I ∈ S_k do
      if %(I_{k,a} − I) < φ then
        if Δ(I_{k,a}, I) > max_rmse then
          rm_img ← I
          max_rmse ← Δ(I_{k,a}, I)
        end if
      end if
    end for
    move rm_img to cluster k + 1
  until no image is removed
  k ← k + 1
until there are no more clusters
```

# 4 Analysis

## 4.1 Run time Analysis

The run time analysis is done in terms of operations per pixel in a single image. For example, if an operation is performed once for each pixel on $n$ images, it would be considered to be performed $n$ times per pixel in a single image.

### 4.1.1 Clustering

The analysis for clustering is performed in two stages. First, the amount of work required to remove *all except* $k$ images from a set of $n$ images using the clustering algorithm is analyzed. The result of this analysis is used to analyze the clustering of $n$ images into $n/k$ clusters of size $k$.

The number of RMSE calculations required is dependent on the properties of the image set. In the worst case, RMSE will be calculated on each image for each iteration of the clustering algorithm. In the best case, RMSE will only be calculated on the one image that is being removed. Both of these cases are unlikely. To simplify the analysis, a few assumptions are made:

1. all clusters have the same size (i.e. $n \bmod k = 0$)
2. The RMSE summations used in this paper represent an "average" case. For this paper, it is assumed that an RMSE calculation is required for each iteration for each image that *will be* removed from the cluster (i.e. $\%(I) < \phi$ for all images that will be removed from the cluster and only for images that will be removed from the cluster).

Let $R_{k,n}$ represent the number of operations that must be performed to remove all but $k$ images from a set of size $n$ (leaving two clusters: one containing $k$ images, and another containing $n-k$ images):

$$\begin{aligned} R_{1,n} &= \overbrace{\sum_{i=1}^{n} i}^{avgs} + \overbrace{\sum_{i=1}^{n} i}^{\%-in} + \overbrace{\sum_{i=1}^{n-1} i}^{RMSE} \\ &= \frac{2n(n+1)}{2} + \frac{(n-1)(n)}{2} \\ &= \frac{2n^2+2n+n^2-n}{2} \\ &= \frac{3n^2+n}{2} \end{aligned}$$

The three summations represent calculations for average images, "percent inlying" values, and RMSE.

To remove all but two images from a set of size $n$:

$$
\begin{aligned}
R_{2,n} &= \sum_{i=2}^{n} i + \sum_{i=2}^{n} i + \sum_{i=1}^{n-2} i \\
&= \frac{2(n-2+1)(n+2)}{2} + \frac{(n-2)(n-1)}{2} \\
&= \frac{2(n^2+2n-2n-4+n+2)+(n^2-n-2n+2)}{2} \\
&= \frac{2(n^2+n-2)+(n^2-3n+2)}{2} \\
&= \frac{2n^2+2n-4+n^2-3n+2}{2} \\
&= \frac{3n^2-n-2}{2}
\end{aligned}
$$

To remove all but $k$ images from a set of size $n$:

$$
\begin{aligned}
R_{k,n} &= \sum_{i=k}^{n} i + \sum_{i=k}^{n} i + \sum_{i=1}^{n-k} i \\
&= \frac{2(n-k+1)(n+k)}{2} + \frac{(n-k)(n-k+1)}{2} \\
&= \frac{2(n^2+nk-nk-k^2+n+k)+(n^2-nk+n-nk+k^2-k)}{2} \\
&= \frac{2(n^2+n+k-k^2)+(n^2+n-k-2nk+k^2)}{2} \\
&= \frac{2n^2+2n+2k-2k^2+n^2+n-k-2nk+k^2}{2} \\
&= \frac{3n^2+3n-2nk+k-k^2}{2}
\end{aligned}
$$

Let $C_{k,n}$ represent the number of operations required to separate $n$ images into clusters of equal size $k$, using the above formula:

$$
\begin{aligned}
C_{k,n} &= \sum_{i=1}^{n/k} \frac{3(ki)^2 + 3(ki) - 2(ki)k + k - k^2}{2} \\
&= k/2 \left( \sum_{i=1}^{n/k} (3ki^2 + 3i - 2ki + 1 - k) \right) \\
&= k/2 \left( \sum_{i=1}^{n/k} 3ki^2 + \sum_{i=1}^{n/k} 3i - \sum_{i=1}^{n/k} 2ki + \sum_{i=1}^{n/k} 1 - \sum_{i=1}^{n/k} k \right) \\
&= k/2 \left( 3k \sum_{i=1}^{n/k} i^2 + 3 \sum_{i=1}^{n/k} i - 2k \sum_{i=1}^{n/k} i + n/k - n \right) \\
&= k/2 \left( 3k \frac{(n/k)(n/k+1)(2n/k+1)}{6} + (3-2k) \frac{(n/k)(n/k+1)}{2} + n/k - n \right) \\
&= k/2 \left( \frac{k(n^2/k^2 + n/k)(2n/k+1) + (3-2k)(n^2/k^2 + n/k)}{2} + n/k - n \right) \\
&= k/2 \left( \frac{k(\frac{n^2+nk}{k^2})(\frac{2n+k}{k}) + (3-2k)(\frac{n^2+nk}{k^2})}{2} + n/k - n \right) \\
&= k/2 \left( \frac{\frac{(n^2+nk)(2n+k)}{k^2} + \frac{(3-2k)(n^2+nk)}{k^2}}{2} + n/k - n \right) \\
&= k/2 \left( \frac{(n^2+nk)(2n+k) + (3-2k)(n^2+nk) + 2nk - 2nk^2}{2k^2} \right) \\
&= 1/2 \left( \frac{2n^3 + n^2k + 2n^2k + nk^2 + 3n^2 + 3nk - 2n^2k - 2nk^2 + 2nk - 2nk^2}{2k} \right) \\
&= \frac{2n^3 + 3n^2 + n^2k + 5nk - 3nk^2}{4k}
\end{aligned}
$$

After the clusters have been calculated, an MST must be calculated for each cluster (each containing $k+2$ images: the images in the cluster, plus the average and zero images for that cluster), including the average cluster (containing $n/k+2$ images):

$$
\begin{aligned}
n/k\binom{k+2}{2} + \binom{n/k+2}{2} &= \frac{n(k+2)(k+1)}{2k} + \frac{(n/k+2)(n/k+1)}{2} \\
&= \frac{n(k+2)(k+1)}{2k} + \frac{(\frac{n+2k}{k})(\frac{n+k}{k})}{2} \\
&= \frac{n(k+2)(k+1)}{2k} + \frac{(n+2k)(n+k)}{2k^2} \\
&= \frac{nk(k+2)(k+1) + (n+2k)(n+k)}{2k^2} \\
&= \frac{nk(k^2+3k+2) + n^2 + 3nk + 2k^2}{2k^2} \\
&= \frac{nk^3 + 3nk^2 + 2nk + n^2 + 3nk + 2k^2}{2k^2} \\
&= \frac{nk^3 + 3nk^2 + n^2 + 2k^2 + 5nk}{2k^2}
\end{aligned}
$$

Therefore, the total number of operations required to cluster $n$ images into clusters of size $k$ and calculate the MST for each cluster is:

$$
\begin{aligned}
T_{k,n} &= \frac{2n^3 + 3n^2 + n^2k + 5nk - 3nk^2}{4k} + \frac{nk^3 + 3nk^2 + n^2 + 2k^2 + 5nk}{2k^2} \\
&= \frac{k(2n^3 + 3n^2 + n^2k + 5nk - 3nk^2) + 2(nk^3 + 3nk^2 + n^2 + 2k^2 + 5nk)}{4k^2} \\
&= \frac{2n^3k + 3n^2k + n^2k^2 + 5nk^2 - 3nk^3 + 2nk^3 + 6nk^2 + 2n^2 + 4k^2 + 10nk}{4k^2} \\
&= \frac{2n^3k + 2n^2 + 3n^2k + n^2k^2 + 11nk^2 - nk^3 + 4k^2 + 10nk}{4k^2}
\end{aligned}
$$

#### 4.1.2 MST$_a$

For MST$_a$, the average image must be calculated, followed by the complete graph on all images plus the zero and average images:

$$
\begin{aligned}
T_{MST_a} &= n + \binom{n+2}{2} \\
&= n + \frac{(n+2)(n+1)}{2} \\
&= \frac{(n+2)(n+1) + 2n}{2} \\
&= \frac{n^2 + 3n + 2 + 2n}{2} \\
&= \frac{n^2 + 5n + 2}{2}
\end{aligned}
$$

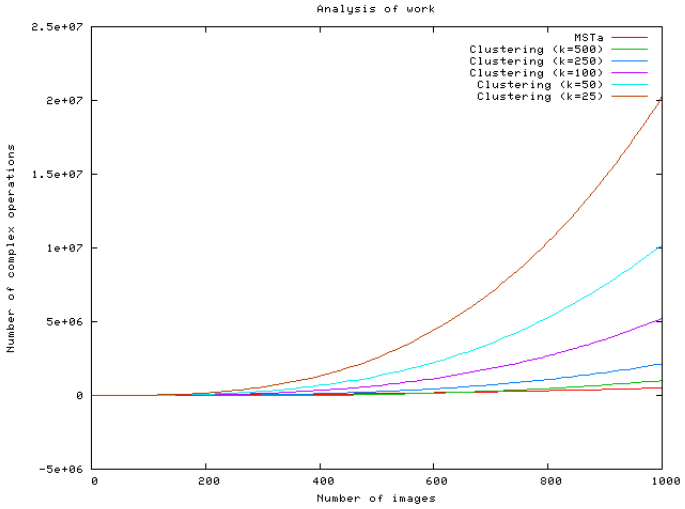


Figure 1: Number of images vs. number of operations.

#### 4.1.3 Comparison

The clustering scheme, as implemented, is slower than the $\text{MST}_a$ scheme. This is clear when the above equations for number of operations based on the number of images in the set is plotted on a graph, as in Figure 1. $\text{MST}_a$ is slower until the number of images in the set exceeds the size of a single cluster. This trend continues for larger cluster sizes. Practically, it does not make sense to cluster the images unless the image set is larger than a single cluster. This means that for practical purposes, the $\text{MST}_a$ scheme is always faster than the clustering scheme.

Additional analysis reveals that most of the computational work is done on the clustering step of the clustering scheme, not the $\text{MST}_a$ step. Further, the computational work involved in simply clustering the images (without running $\text{MST}_a$ on the clusters) exceeds that of performing the $\text{MST}_a$ algorithm on the entire set of images (see Figure 2). As expected, the computational work required to run the $\text{MST}_a$ algorithm on the clustered image set is less than that required to run the $\text{MST}_a$ algorithm on the unclustered set (see Figure 3).

## 5 Experimental Results

Preliminary experiments have been conducted on the two image sets—the combination set and the Joe image set [3, 4]. Results for the combination set can be seen in Figure 4, and results for the Joe set can be seen in Figure 5, which plot bitrate against average distortion. For these experiments, we coded the image sets using each set mapping scheme at varying bitrates, and measured the distortion of the reconstructed image using RMSE for each image in the set

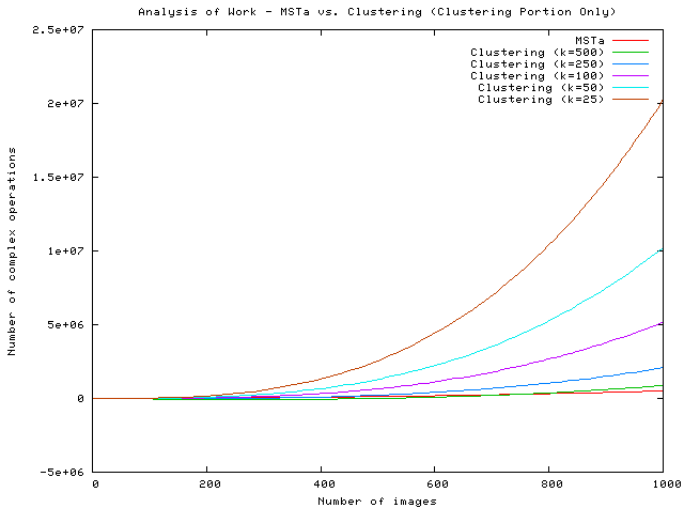


Figure 2: Number of images vs. number of operations: clustering portion of clustering algorithm only.

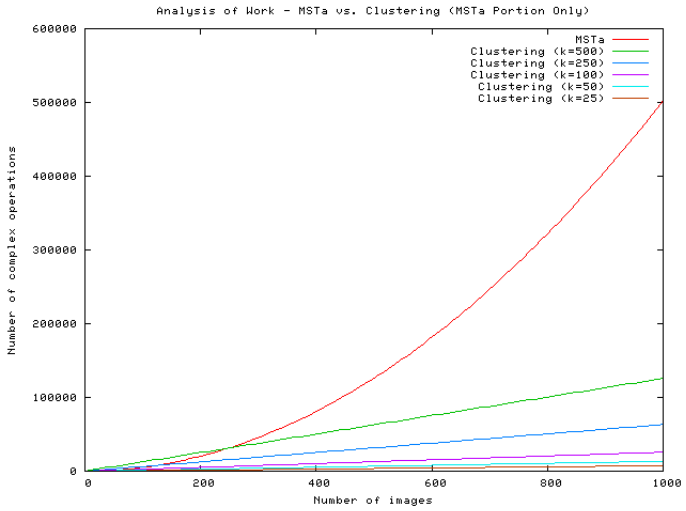


Figure 3: Number of images vs. number of operations: $\text{MST}_a$ portion of clustering algorithm only.

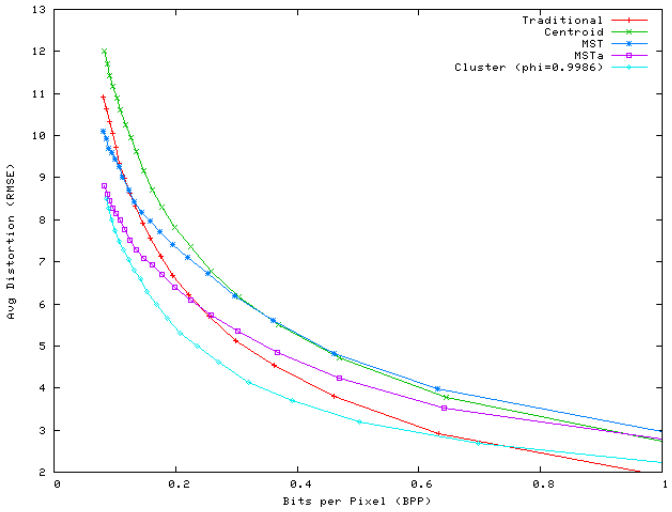


Figure 4: Results for the combination image set.

at each bitrate. Therefore, a curve that is lower and to the left represents better compression performance. For both image sets, the HMST$_a$ scheme outperforms the other set mapping strategies.

## 5.1 Explanation of Results

### 5.1.1 Combination

The combination image set contains 29 images from the pig image set and 28 images from the Galway image set [3, 4]. Clearly, the images form two tight, distinct clusters (the images in a single cluster are quite similar to each other, but quite dissimilar to the images in the other cluster. See Figures 6 and 7).

With the other set mapping schemes such as centroid and MST$_a$, only one average is formed for the entire set. In the case of the combination set, the average image contains elements of both the pig and Galway images, and is not a good predictor for any image in the set. Figure 8 shows the average image from the application of MST$_a$ on the combination set. This shows that the prediction using the average image is poor.

With the HMST$_a$ strategy, the two clusters are identified and separated, and an average image is calculated for each cluster. The average images are much better predictors for the images in the clusters, because they only contain elements from a set of similar images. As a result, the difference images are easier to compress. See Figure 9 for a sample cluster average image from the HMST$_a$ algorithm.

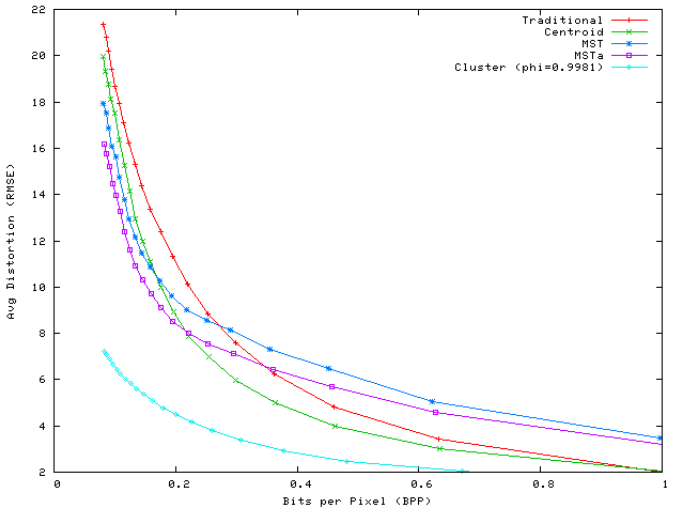


Figure 5: Results for the Joe image set.

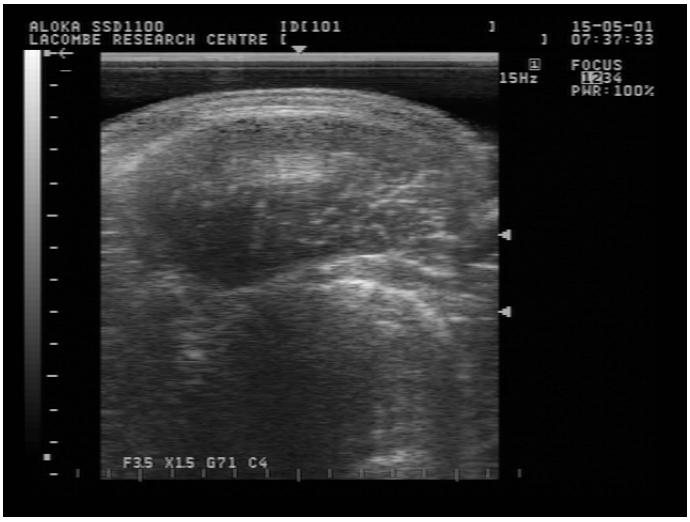


Figure 6: Sample image from the pig set.


Figure 7: Sample image from the Galway set.


Figure 8: Average image from $\text{MST}_a$ on the combination set.

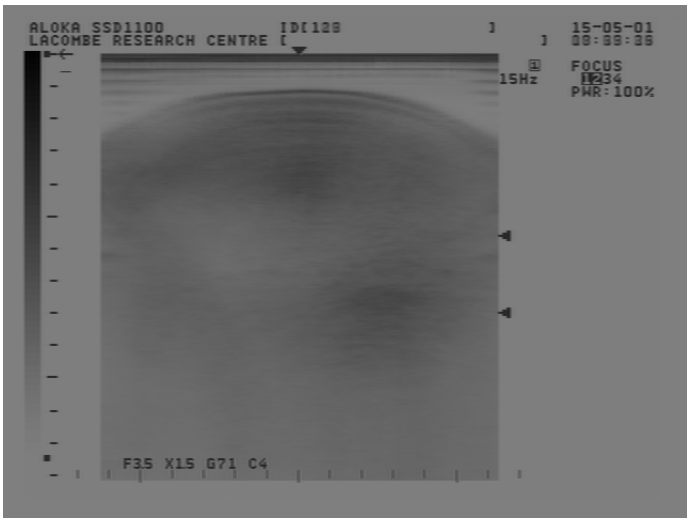


Figure 9: Sample average image from $\text{HMST}_a$ on the combination set.


Figure 10: Sample images from the Joe set.

### 5.1.2 Joe

The Joe image set contains time lapsed photographs of an outdoor scene, captured from a webcam. They are intended to allow the viewer to see the weather in Victoria, B.C., so most of the image is taken up by sky. The images used in these experiments were captured at different times throughout the day, and in different weather conditions, so the sky portion of the images is significantly different among the images. See Figure 10 for sample images. The drastic variance in the sky portion of the images has a strong affect on the average image, and as a result, the average image is a poor predictor for the images, and the difference images contain a wide range of pixel values. The average image of the entire set is shown in Figure 11, and a sample difference image from the $\text{MST}_a$ scheme is shown in Figure 12. The $\text{HMST}_a$ strategy performs well on the Joe set for reasons similar to why it performs well on the combination set. Images in similar clusters show similar sky conditions. This means that the average im-


Figure 11: Average image from $\text{MST}_a$ on the Joe set.

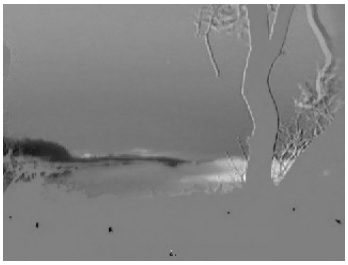

Figure 12: Sample difference image from $\text{MST}_a$ on the Joe set.

age for each cluster will be a better predictor for images in that cluster, which means that difference images are easier to compress. Figure 13 shows a sample difference image from the application of $\text{HMST}_a$ on the Joe image set.

# 6 Conclusion

The compression performance of the $\text{HMST}_a$ strategy is better than that of the traditional, centroid, MST, and $\text{MST}_a$ strategies on image sets that contain multiple tight clusters. This is at the computational expense of running a clustering algorithm on the images prior to applying the $\text{HMST}_a$ algorithm. In

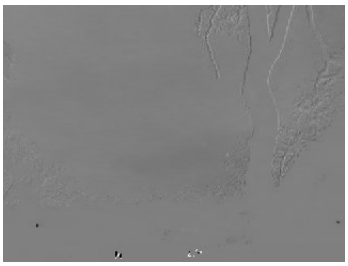

Figure 13: Sample difference image from $\text{HMST}_a$ on the Joe set.

some cases, this computational expense may be avoidable if some outside information about the images is available. For example, if it is known that certain images are photographs of certain objects, the clustering step may be avoided.

Future experiments will be conducted to determine how well $\text{HMST}_a$ performs for image sets that do not contain tight clusters, and to gauge the impact of using wavelet packet coding rather than JPEG2000 to compress the difference images. Future work may also include the application of different clustering algorithms before the application of the $\text{HMST}_a$ scheme.

# References


[1] C.-P. Chen, C.-S. Chen, K.-L. Chung, H.-I. Lu, and G. Tang. Image set compression through minimal-cost prediction structures. In *IEEE International Conference on Image Processing*, pages 1289–1292, 2004.

[2] The JPEG Committee. JPEG 2000. `http://www.jpeg.org/jpeg2000/index.html`.

[3] B. Gergel. Automatic compression for image sets using a graph theoretical framework. Master's thesis, University of Lethbridge, 2007.

[4] B. Gergel, H. Cheng, C. Nielsen, and X. Li. A unified framework for image set compression. In *Proceedings of the 2006 International Conference on Image Processing, Computer Vision, & Pattern Recognition*, pages 417–423, 2006.

[5] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, 1993.

[6] K. Karadimitriou. *Set Redundancy, the Enhanced Compression Model, and Methods for Compressing Sets of Similar Images*. PhD thesis, Louisiana State University, 1996.

[7] K. Karadimitriou and J. M. Tyler. The centroid method for compressing sets of similar images. *Pattern Recognition Letters*, 19(7):585–593, 1998.

[8] C. Nielsen and X. Li. MST for lossy compression on image sets. In *Proceedings of the Data Compression Conference*, page 463, 2006.

[9] C. Nielsen, X. Li, and K. Abma. Methods of grouping similar images for compression coding. In *Proceedings of The 2005 International Conference on Computer Vision*, pages 93–99. CSREA Press, 2005.