

# Interactive Visualization and Tuning of SIFT Indexing

Pavan Kumar Dasari<sup>1</sup> and P. J. Narayanan<sup>1</sup>

<sup>1</sup> International Institute of Information Technology, Hyderabad

---

## Abstract

*Indexing image data for content-based image search is an important area in Computer Vision. The state of the art uses the 128-dimensional SIFT as low level descriptors. Indexing even a moderate collection involves several millions of such vectors. The search performance depends on the quality of indexing and there is often a need to interactively tune the process for better accuracy. In this paper, we propose a visualization-based tool to tune the indexing process for images and videos. We use a feature selection approach to improve the clustering of SIFT vectors. Users can visualize the quality of clusters and interactively control the importance of individual or groups of feature dimensions easily. The results of the process can be visualized quickly and the process can be repeated. The user can use a filter or a wrapper model in our tool. We use input sampling, GPU-based processing, and visual tools to analyze correlations to provide interactivity. We present results of tuning the indexing for a few standard datasets. A few tuning iterations result in an improvement of over 4% in the final classification performance, which is significant.*

Categories and Subject Descriptors (according to ACM CCS): H.3.3 [Information Storage and Retrieval]: Information search and retrieval—clustering H.5.2 [Information Interfaces and Presentation]: User Interfaces—graphical user interfaces

---

## 1. Introduction

CBIR is the process of retrieving desired images from a large collection based on syntactical image features. Recent advancements in computer vision has introduced several new low-level features like Scale Invariant Feature Transform (SIFT) [Low99] to describe images other than color, texture, shape, making it possible to achieve significant results in image retrieval and classification. SIFT descriptors are computed either at interest points [Low01] or in a uniform grid [DT05] in a 128 dimensional space. A small image in general is described by a few hundreds of such descriptors. Hence, for a reasonably sized collection of a few thousands of images, the volume of data points explodes to a few million.

CBIR draws parallel with text search techniques. In document retrieval, each document is represented by a vector using bag of words representation. Analogous to this approach, for CBIR, each image is described by an image vector which is a histogram of *visual words*. Since each SIFT descriptor is a low-level feature, the entire set of descriptors extracted from the image collection is divided into a fixed number of clusters with each cluster center denoting a visual word

(For detailed explanation, refer to [SZ03]). SIFT descriptors in each cluster are similar in some sense of objectivity. To improve the retrieval accuracy using image vectors, Li et al. [LP05] incorporate learning methods like Support Vector Machine [TC01]. However, retrieved results might not always convey the information required to boost learning parameters. In such a case, there is a need to fall back on information already available and organize it much better.

In this paper, we focus on overall indexing quality by enabling the user with an interface to analyze the match quality. In contextual terms, clusters formed from SIFT vectors must be judged if they form a relevant visual word. We believe that there is a need to provide an interactive feedback based interface to control the indexing process which can improve the overall quality. We propose a visualization-based framework and a tool to help tune indexing systems for CBIR. We use a feature selection approach to improve the quality of clustering of SIFT vectors by weighing each dimension differently. Weights can be set interactively with automatic suggestions. The tool supports both filter and wrapper model of clustering. It also provides an interactive interface to analyze the clusters formed, by using a graph visualization scheme

for the cluster centres as well as the vectors in each cluster. Relative cluster validity techniques are implemented and visualized as a line chart to assist a user understand the quality of clusters formed. To make the process interactive, a GPU is used for fast clustering as well as to compute the graph layout. We observe a classification accuracy of 56.6% overall using our tool for UIUC dataset [03] consisting 15 different categories. Our main contribution is the combination of interactive visualization techniques to improve the image indexing problem. Though the tool is specifically tuned for SIFT based indexing, it can be used for many learning-based problems that use high dimensional vectors.

## 2. Background and Related Work

A variety of visualization schemes have been proposed to display image result collections. Chen et al. [CGR00] cluster images based on low level features like color, shape. Imo et al. [IKH08] provide an interactive method to visualize color histograms and texture features of images. Nakazato et al. [NH01] visualize retrieved images in two or three dimensional space with each dimension denoting some characteristic of feature space. To make characteristics of the feature space more apparent, some approaches map high dimensional space down to a number of depictable dimensions, for example, multidimensional scaling [RT01] or principal component analysis [Jol02]. Though the information loss in these approaches is apparent, they allow an abstract global view of the information. These methods do not scale well for collections of thousands of images. More over, they allow only to visualize retrieved results but provide no interface for user feedback.

Interactive relevance feedback techniques incorporate human perception subjectivity into retrieval process. They provide users with an opportunity to evaluate retrieved results and automatically refine queries on the basis of their perception [RH99] [SZM00]. Zhong et al. [SLZ01] use PCA to reduce the dimensionality of feature space and define a method to extract features from positive images provided by relevance feedback. Dimensionality reduction techniques are used for two main reasons: better cluster quality and reduced data size. Various methods have been proposed to identify low dimensional subspaces, which can be broadly classified into *feature selection* and *feature transformation* algorithms. Methods based on variance (such as principal components analysis) need not select good features for clustering, as features with large variance can be independent of the intrinsic grouping of the data. Feature selection algorithms can be divided into two categories [BL97], [KJ97]: *filters* and *wrappers*. Most of the dimensionality reduction methods lack user interaction in deciding the final feature space. The Weka system [HFH\*09] provides implementations to many of these techniques but the user is allowed only to choose parameter values before executing the algorithms. Previous investigations on automated feature selection for

unsupervised learning reveal that no single feature selection criterion is best for every application. Dy et al. [DB00] incorporate scatterplots and user interaction to guide feature subset search and enable a deeper understanding of the data. But this is a very specific implementation of a wrapper technique and is not scalable to large datasets.

While we look to reduce the number of dimensions, it is useful to analyze individual feature correlations. Relationships between two dimensions can be best visualized in a scatterplot. A Multidimensional dataset can be visualized using a scatterplot matrix [JMCT83]. But however, this technique has got its own disadvantages. Given the limited size of plotting region in most display media, size of each panel must be reduced accordingly. At some point, visual resolution of point clouds within panels will degrade to an unacceptable degree. Further, it cannot really show multivariate structure, because scatterplot within each panel is constructed completely independent of information in any other panel. PCP [ID90] on the other hand provides support to study geometry of the data. In this method, each dimension corresponds to an axis. N axes are organized as parallel lines, typically vertical with uniform spacing. A vector  $V$  with values  $(V_1, V_2, \dots, V_n)$  is visualized as a polyline connecting points  $(U_1, U_2, \dots, U_n)$  on N vertical axes. A number of extensions to PCPs exist, like hierarchical [FWR99] and multiresolution methods, 3D PCPs [JCJ05] and combinations of clustering, binning [AdOL04] and other features like outlier detection [JLJC05] to reduce visual clutter there by supporting large datasets.

Since, there are no predefined classes and no examples that would show what relations are valid among the data, the final partition of a dataset requires some method of evaluation to understand the underlying quality. Halkidi et. al. classify cluster validity methods into three categories, namely internal, external and relative measures [HBV02]. Internal and external approaches are based on statistical tests and their major drawback is their high computational cost. Moreover, the indices related to these approaches aim at measuring the degree to which a data set confirms an a-priori specified scheme.

## 3. SIFT Tuning Framework

Few attempts have been made to incorporate more user subjectivity into the summarization of low-level features. One of the key contributions of this paper is the redefinition of CBIR process by incorporating a visual framework to generate user-favoured clusters of low-level features like SIFT extracted from each of the images. A learning based CBIR system relies on the quality of image vectors and relevance feedback provided by the user to train the classifier. The visual words should be of high quality. Generating qualitative clusters is possible if the user identifies the behaviour of underlying data points and controls the entire process interactively.

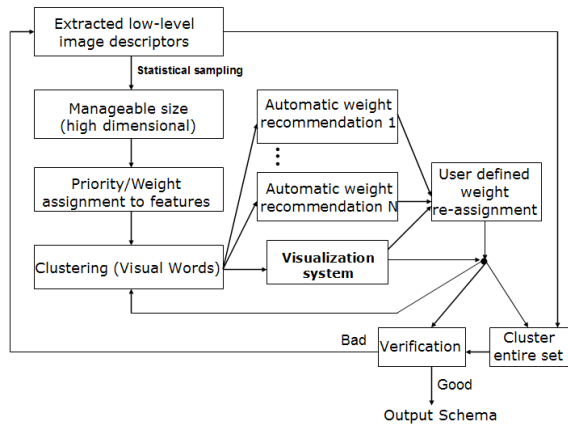


Figure 1: Framework: Overview of the procedure

Visualization of large amounts of high dimensional data has been an active research area in information visualization. Keim [Kei02] gives a good overview and categorization of relevant visualization techniques. Many of these methods aim at the identification of interesting formations in the data, such as linear correlations. However, to analyze SIFT descriptors extracted over a collection of images, these do not apply directly. Thus, we aim at presenting a visualization based framework with an implemented tool to tune the entire process of indexing SIFT.

Interactive analysis of such huge abstract datasets is not possible using individual traditional methods. We sample the entire dataset to a manageable size using stratified sampling method. We follow a feature selection approach and incorporate a rank-by-feature schema to identify subspaces. Distribution along each of the dimensions can be analyzed by its corresponding histogram and box-plot. We use a Parallel Coordinate Plot widget to enable a user identify two-dimensional correlations. Once dimensions are analyzed, suitable weights are chosen to generate visual words. As the framework suggests, a user is free to choose either a filter or a wrapper model of clustering which can be incorporated as a plugin in the tool. Whatever might be the method used to compute visual words, it is necessary to analyze the cluster structure formed. Graph layout provides an excellent means of analyzing such high dimensional structures. Since it is an unsupervised clustering process, users must be able to evaluate the resulting clusters formed using some qualitative measure. Relative index measures form an excellent choice. User can interactively re-assign weights to each feature vector based on his observation of a clustering process. Some automatic methods of suggesting weights have also been included to aid the choice. The procedural framework after we sample a dataset is described as follows.

### 3.1. Feature Selection and Weight Assignment

In an exploratory analysis, users do not know in advance what they are looking for or what kind of projections are interesting. In such a case, it is necessary to provide tools to easily identify interesting projections. We incorporate a rank-by-feature approach proposed by Seo et al. [SS05]. In this framework, users can incorporate their interests into an interactive exploratory analysis process by selecting a ranking criterion among available ranking schemas. All possible axis-parallel projections are ranked by the criterion function.

We integrate four ranking criteria into our tool, since they are common and fundamental measures for distribution analysis.

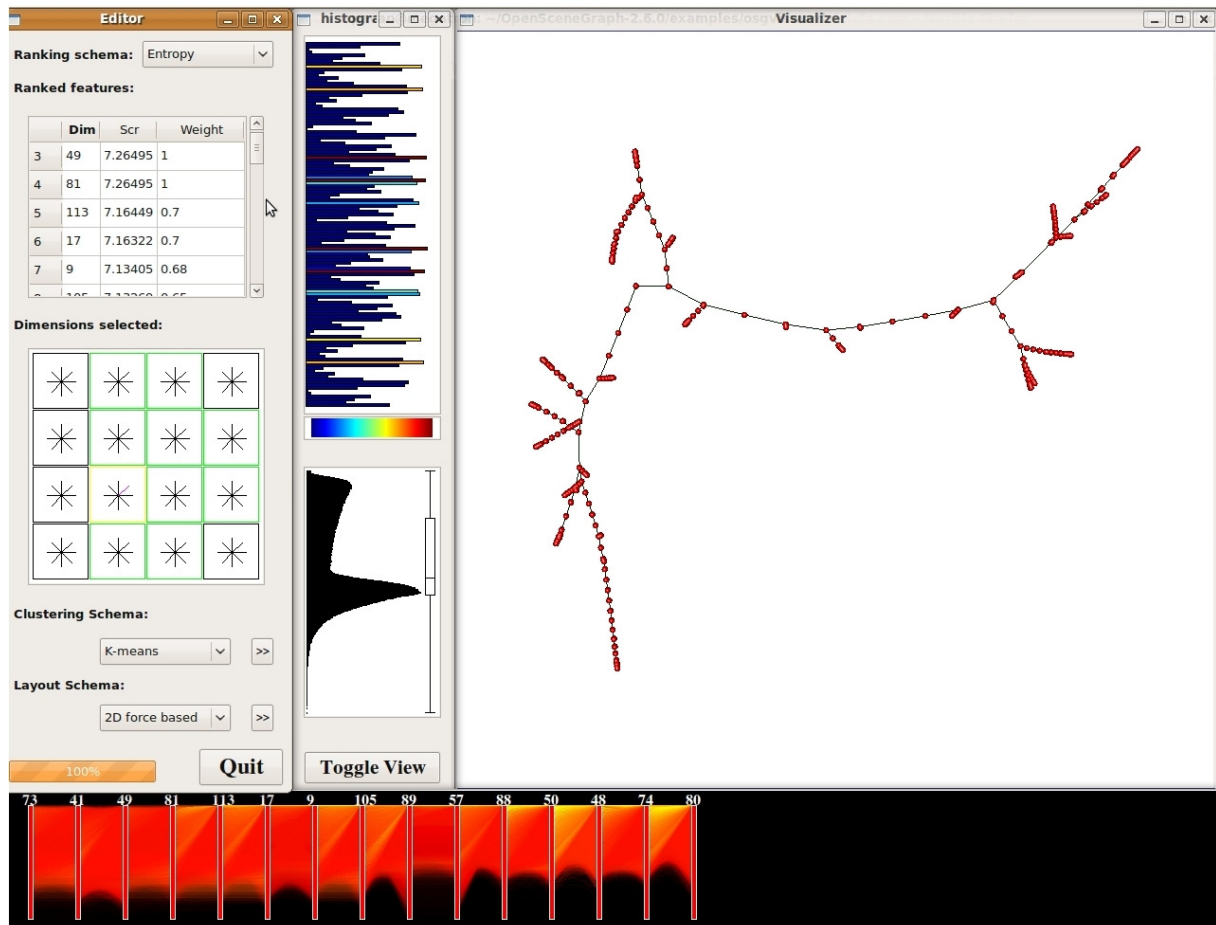
- Entropy of the distribution (0 to  $\infty$ )
- Normality of the distribution (0 to  $\infty$ )
- Number of potential outliers (0 to N)
- Number of unique values (0 to N)

We use an entropy measure to compute uniformity. Given  $k$  bins in a histogram  $H$ , its entropy  $E$  is defined as  $E(H) = - \sum_{i=1}^k P_i \log_2(P_i)$  where  $P_i$  is the probability that an item belongs to  $i$ -th bin. We chose *omnibus moments test* for normality from several statistical tests available. Several outlier detection algorithms have been proposed in the field of data mining [Pet03]. We select an item of value  $d$  to be an outlier if,  $d > (Q_3 + 1.5 * IQR)$  or  $d < (Q_1 - 1.5 * IQR)$  where IQR is the interquartile range (defined as the difference between the first quartile ( $Q_1$ ) and the third quartile ( $Q_3$ )).

We can however notice that any filter model based feature selection method can be incorporated instead of a ranking schema and assign uniform weights to subset produced.

#### 3.1.1. One-dimensional Distribution Analysis

Users may begin their exploratory analysis by scrutinizing each dimension one by one. A mere look into the distribution of values of a dimension gives useful insights. The tool provides *histograms* and *boxplots* for graphical display of 1D data as shown in Fig. 2. Histograms graphically reveal the skewness and scale of the data. Boxplots provide a quick way of examining one or more sets of data graphically by showing a five-number summary (the minimum, the first quartile, the median, the third quartile, and the maximum). These numbers provide an informative summary of dimension's center and spread and thus help in selecting dimensions for deriving a model. The sub-interface consists of four coordinated parts: the rank box, the table view, the order view and the histogram view. Users can select a ranking criterion from the rank box, which is a combo box, and then see the overview of scores for all dimensions in the table view (Figure 2) according to the ranking schema selected. All dimensions in the data table are sorted in decreasing value of scores on default. The table view consists of three columns. The first column denotes the dimension name, second denotes the score of that dimension according to rank-



**Figure 2: Sift-visualizer:** It consists of a 1D ranking framework providing details about distribution of each dimension as a histogram combined with box plots. Weights can be assigned to each dimension either directly in the table view or selecting appropriate orientation angle in the glyph view. User can select appropriate clustering schema and layout method. Its graph is displayed in the Visualizer window. Dimensions which are assigned weights, are rendered in the PCP view denoting correlations between two adjacent dimensions.

ing schema, and the third column in the table view displays weight assigned to it by user.

The order view is a bar chart, where in the length of the bar denotes the rank of that dimension. All dimensions are aligned from top to bottom in the original order and each dimension is color coded by corresponding weight-value (third column in the table view). Weight is directly proportional to the color scale which is displayed below the order view. Its mapping can be obtained by a simple mouse hover over the display bar (Fig. 2). A mouse hover over a bar in the order view displays the corresponding dimension *id* in a tooltip window. Thus, user can preattentively identify dimensions of highest and lowest rank and observe the overall pattern. The user is provided an option to interchange the parameters in

the order view such that color of each bar denotes the rank and length, its weight. The histogram view is a combination of one-dimensional histogram plot and box-plot to display 1D projections.

The mouseclick event in the rank view or a cellclick event in the table view is instantaneously relayed to other views. The corresponding item is highlighted in the rank and the table view, and its histogram and box plot is rendered in the histogram view. In other words, a change of dimension in focus in one of the rank or table view leads to instantaneous change of dimension in focus in other component views.

### 3.1.2. Identifying Correlations

For better exploration of unsupervised, multi-dimensional data, after scrutinizing one-dimensional projections, it is natural to move on to two-dimensional projections where pairwise relationships can be identified. Based on our discussion in section 2, we build on the idea of bin map and perform refinements to display the structure of SIFT descriptors.

PCPs have little to gain from high precision floating point representation of data. When data values are rounded to a lower precision representation, the maximum erroneous displacement that a line in a plot will have is directly related to the rounding error. The axes of PCP currently are less than 127 pixels. Hence, a quantization to 8-bit values will yield a maximum displacement of a single pixel which doesn't show any effect in the current scenario. This step reduces the data from 4 bytes to a single byte per point per attribute, greatly reducing the necessary storage.

A PCP without any selections can be quickly generated solely from the joint-histograms of the data. We make use of binning approach proposed by Artero et. al. [AdOL04]. Only the joint histogram between each pair of neighbouring axes is needed to build the parallel coordinate plot using this technique. Fast exploration of data is made possible by computing joint histograms over all pairs of axes. For  $N$  axes, we get  $\frac{N(N-1)}{2}$  histograms for all pairs.

We adopt the rendering approach of Philipp et. al. [MKO\*08] where histogram bins form a direct basis for drawing the primitives. Instead of having to draw a line for each data point, only a single primitive is drawn for each histogram bin. We use additive blending to combine all drawn primitives. We use a square-root intensity scale, as to prevent over-saturation of high-density areas in the plot, while keeping a good visual contrast in low intensity areas.

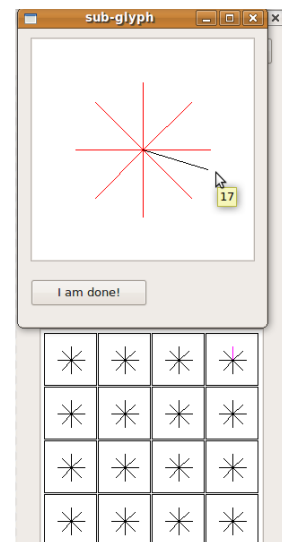
A value change event in the weight column in the table view is instantaneously relayed to PCP display. We believe that displaying joint histograms in the order of precedence of selected dimensions will show correlations which might be useful to re-define importance to each of the dimensions. If a weight  $W (>0)$  is assigned to a dimension by the user, the corresponding joint histogram is rendered in order of precedence of selection. For example, from Figure 2, say dimension 113 is assigned a weight, then a joint histogram between 81 and 113 is rendered. Next, when dimension 17 is given some weight, a bi-histogram between 113 and 17 is loaded into memory and rendered in PCP display.

Thus, we provide a user with the power to analyze multivariate structure of *interesting* dimensions in 1D projections. User might find 1D projection of a dimension to be interesting, but it might not have any correlation with other dimensions of significant interest. In such a case, user can revert back to remaining dimensions by deselecting the uncorrelated dimension. De-selecting a dimension can be performed

simply by reassigning its corresponding weight  $W$  to zero in the table view.

### 3.1.3. Weight Assignment Using Glyphs

Since we consider scale invariant feature transform descriptors, the number of dimensions present are 128. If we observe how SIFT is computed, we notice that each interest point (which is computed using some interest point detector like [MS04]) is divided into a  $4 \times 4$  matrix where cell size depends on the scale computed by interest point detector. Eight orientation angles are chosen describing the information in each cell. Following a similar approach, we display a  $4 \times 4$  matrix of glyphs where each glyph is further divided into eight orientations as shown in Figure 2. [vA02] use glyphs to display information about k-dimensional data.



**Figure 3: Orientation view:** Visually assigning weights to adjacent dimensions. Greater the inclination, more the weight given to an orientation.

A Mouseclick event in the glyph view generates a zoomed-in version of the corresponding cell in which mouseclick event occurred. User can visually assign weights between neighbouring dimensions as shown in Figure 3. If a weight is assigned to a dimension, its corresponding cell in the  $4 \times 4$  cell matrix is highlighted in green. This mode of assigning weights is very useful if two neighbouring dimensions are of sufficient interest and user can visually approximate priorities. A left mouseclick event in the orientation view assigns a weight to each of the corresponding adjacent dimensions with respect to the angle selected. One method is to assign a weight relative to cosine of the angle of selection. A right mouseclick deselects previously assigned weights in the cell. The updated weights are relayed to the table view on occurrence of either of the events.

### 3.2. Data Clustering

After potentially identifying a weighted sub-space, user clusters the set of data points. Since different users might be interested in different clustering methods, it is desirable to allow users to customize the available set of clustering schemas. However, we have chosen the standard  $k$ -means [JD88] as a starting point and implemented it on the GPU using CUDA [06] to achieve significant speed up. A modified Euclidean distance schema is incorporated into computing distance between cluster means and points. Distance between a cluster mean  $P$  and a data point  $Q$  is computed using the formula  $D_{(pq)} = \sqrt{\sum_{i=1}^{128} W_i * (P_i - Q_i)^2}$ , where  $W_i$  ( $0 \leq W_i \leq 1$ ) denotes weight assigned to dimension  $i$  in the table view. After every iteration, we update cluster means as in the standard procedure.

### 3.3. Visualization System for Cluster Analysis

#### 3.3.1. Graph Layout

Often, clustering algorithms go hand in hand with graph drawing methods providing important means of dealing with increasingly large datasets. A good layout effectively conveys the key features of a complex structure or system to a wide range of users. The primary goal of these types of methods is to optimize the arrangement of nodes such that strongly connected nodes appear close to each other.

The user is presented with a two-dimensional representation of multi-dimensional data, that is easy to understand and can be further investigated. We make use of Euclidean Minimal Spanning Tree (EMST) proposed by Stuetzle [Stu03]. This skeleton forms the basic layout representation of the data. In order to compute the EMST, we need the high-dimensional point cloud in attribute space spanned by the entire set of attributes. Hence, for each position in the data set, an attribute vector that consists of individual multi-variate values is computed. A spanning tree connects all points in attribute space with line segments such that the resultant graph is connected and has no cycles. For a graph with  $n$  nodes, we end up with  $n - 1$  edges in the spanning tree. A spanning tree is an EMST if the sum of the euclidean distances between connected points is minimum. Since our graph layout follows a drill-down approach, each graph contains only a few thousands to tens of thousand of nodes. We apply Prim's algorithm [CT76] to compute the minimal spanning tree. For a graph with  $N$  nodes,  $\frac{N(N-1)}{2}$  edges are chosen where each edge is given a weight by finding the euclidean distance between the pair of points. Our current approach is based on graph drawing, where the EMST is first projected to 2D by assigning each node an arbitrary position. Afterwards, the graph is laid out to achieve appropriate edge lengths and few edge intersections.

We choose the Fast Multipole Multilevel Method (FM3) method since it produces pleasing layouts and is relatively

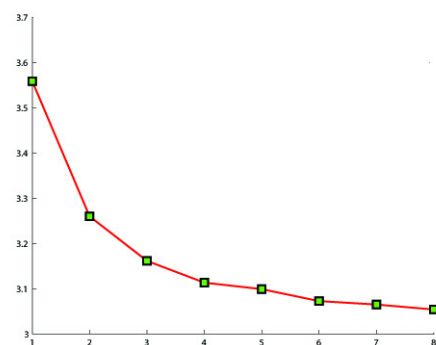
fast [HJ05]. The basic approach of this algorithm tries to coarsen recursively an input graph  $G_0$  to produce a series of smaller graphs  $G_1 \dots G_k$ , until the size of coarsened graph falls below some threshold. We use the GPU implementation of a modified version of this algorithm by Godiyal et al. [GHGH09]. Only the multipole expansion coefficients are considered and not the local expansion coefficients to approximate repulsive forces. These coefficients alone are sufficient to produce a high quality layout.

#### 3.3.2. Cluster Validity

Since we use an unsupervised approach of clustering, its often necessary to quantitatively evaluate the final partition. Cluster validity approaches based on relative criteria aim at finding the best clustering scheme that a clustering algorithm can define under certain assumptions and parameters. Here the basic idea is the evaluation of a clustering structure by comparing it to other clustering schemes which were produced by the same algorithm using different weight assignments. In this framework, we provide user an option of choosing from three indices. Namely,

- Davies-Bouldin index
- R-squared index
- SD validity index

For an in-depth description of indices, please refer to [HBV02]. As these indices are relative, we plot each of them as a line graph. Based on the index chosen, user searches for an optimum value denoting the best quality of clusters achieved over the process. Figure 4 denotes a plot of Davies-Bouldin index obtained for one of the experimental setups.



**Figure 4:** Davies-Bouldin index: With each iteration, the value decreases meaning better cluster quality

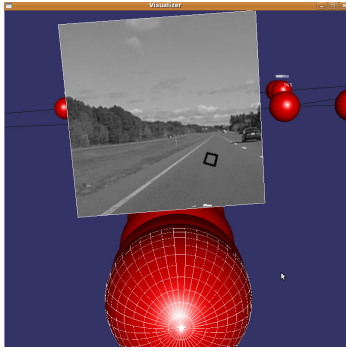
#### 3.3.3. Interaction for Cluster Analysis

A zoom user interface allows user to change the scale of the viewer area. Holding the right mouse button down, a mouse

left and a right makes the graphical display zoom out and zoom in respectively. This helps user pin down to a point of his/her interest.

User can rotate the display on either of the spatial axes (X, Y or Z) to change the view point. This is achieved by holding down the left mouse button and moving in any direction. User can translate the canvas by pressing middle mouse button and making a move in any direction. This provides great flexibility to analyse the structure of the graph layout and pin down to the interest area.

User can move the cursor over any valid data to browse through the details. This is a drill-down approach where visual words are displayed initially according to a graph layout algorithm. User can point the cursor over any node (each node represents a visual word) and click on it to generate a graph layout of underlying SIFT vectors assigned to that particular visual word. Another click on any node denoting a sift descriptor displays its interest region in an image as shown in Figure 5.



**Figure 5:** Drill-down to a sift descriptor in a cluster. The interest region is denoted by a black colored rectangle and the selected node is highlighted with a wired mesh enclosing it.

### 3.4. Automatic Weight Recommendation

It might often be tedious for a user to re-assign weights to each dimension based on cluster analysis. An automatic weight suggestion scheme proves handy in manual weight assignments. There are different ways of suggesting weights based on partition obtained from the clustering method. One way is to choose

$$w_j = \begin{cases} 0 & \text{if } D_j = 0 \\ \frac{1}{\sum_{i=1}^h \left[ \frac{D_j}{D_i} \right]^{\frac{1}{h-1}}} & \text{if } D_j \neq 0 \end{cases}$$

Where  $w_j$  is the attribute weight of  $j$ -th dimension,  $0 \leq w_j \leq 1$  or  $w_j > 1$ ,  $h$  is the number of variables where  $D_j \neq 0$  and

$$D_j = \sum_{l=1}^k \sum_{i=1}^n u_{i,l} d(x_{i,j}, z_{l,j})$$

$$\begin{cases} u_{i,l} = 1 & \text{if } \sum_{j=1}^m w_j d(x_{i,j}, z_{l,j}) \leq \sum_{j=1}^m w_j d(x_{i,j}, z_{t,j}) \\ & \text{for } 1 \leq t \leq k \\ u_{i,l} = 0 & \text{for } t \neq l \end{cases}$$

$Z = \{Z_1, Z_2, \dots, Z_k\}$  is a set of  $k$  vectors representing the centroids of the  $k$  clusters,  $X = \{X_1, X_2, \dots, X_n\}$  is a set of  $n$  objects, each object  $X_i = (x_{i,1}, x_{i,2}, \dots, x_{i,m})$  is characterized by a set of  $m$  dimensions,  $U$  is a  $n \times k$  partition matrix,  $u_{i,l}$  is a binary variable and  $u_{i,l} = 1$  indicates that object  $i$  is allocated to cluster  $l$ ,  $d(x_{i,j}, z_{l,j})$  is the distance or dissimilarity measure between object  $i$  and centroid of cluster  $l$  on the  $j$ -th variable. This method of suggesting weights is a modified version from [HNRL05] where they aim to minimize an objective function

$$P(U, Z, W) = \sum_{l=1}^k \sum_{i=1}^n \sum_{j=1}^m u_{i,l} w_j d(x_{i,j}, z_{l,j})$$

It is only a support process and the final decision to choose weights for each dimension is left to the user.

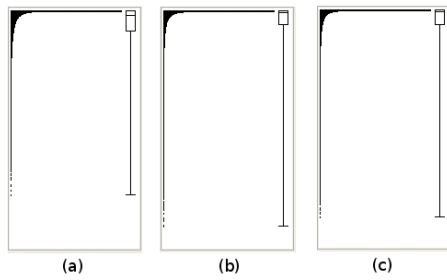
## 4. Experimental Results

We show an application example of the framework with a collection of image classification categories. This dataset contains 4485 images with 15 different classes available with UIUC [03]. We apply SIFT feature detector on all images using Vedaldi's implementation [05]. We obtain close to 1.1 million SIFT vectors for this collection. We implemented the current tool using Qt, OpenSceneGraph [07] and OpenCV. All computations are performed on a 2.4Ghz quad core system with 2.5GB RAM and a GTX 280 graphics card. We use intersection kernel based Support Vector Machine (SVM) to build a classifier and a fast intersection kernel for testing [MBM08]. We sample 60 SIFT descriptors from each image resulting in a total sample size of 0.26 million vectors approximately.

After a series of iterations of weight re-assignment and clustering in tandem over sampled data with manual intervention, we observe that clusters formed are relatively stable, backed by a cluster validity index (figure 4). Using the observed weights, we run a weighted kmeans on the entire dataset with respective  $K$  cluster centers.

Over several loops of above steps, we observe that dimensions, especially those belonging to the corner cells and corresponding to orientations  $135^\circ, 215^\circ, 270^\circ$  are assigned low weights by automatic suggestion schemas. Since we observe lower suggestive weights for a few dimensions in the previous iterative process, we redo the entire process once again. But this time, we discard a set of dimensions  $D_s = \{4, 12, 22, 43, 44, 54, 55, 71, 78, 79, 83, 84, 110, 116\}$  resulting around 11% decrement in data size. The observed overall classification accuracy in terms of percentage is shown in table 1.

Our CUDA based k-means algorithm takes half a second



**Figure 6:** 1D histogram corresponding to dimensions (a)84, (b) 110, (c) 124

VWs	EW	IW	IW- $D_s$
400	51.7	53.7	54.1
500	51.5	53.0	53.7
600	52.1	54.1	54.7
700	52.6	54.3	55.6
800	53.1	54.6	55.9
900	52.8	53.9	55.3
1000	53.4	55.2	56.6

**Table 1:** VW = Number of visual words, EW = K-means using uniform weights, IW = K-means with weights adjusted interactively, IW- $D_s$  = K-means with  $D_s$  dimensions given a weight zero and weights of other dimensions adjusted interactively.

for an iteration for above sampled data. k-means is run upto convergence or 25 iterations, which ever is the earliest, there by consuming just over 12 seconds in a user interaction loop. Initial weight to each of the dimensions is assigned based on the behaviour observed in the histogram and PCP views. Once clusters are computed in a user interaction loop, automatic weights are suggested with  $\alpha = 7$  which takes less than half a second using a CUDA based implementation of method described in section 3.4. We consume about a minute to analyze the cluster quality and interactively adjust weights and proceed to the next round of clustering. In the above experiments, we loop over weight re-adjustment and clustering process eight times after which a considerable drop in Davies-Bouldin index was not observed. After we conclude with a set of corresponding weights based on the sample data, our weighted k-means method is run over the entire dataset as an offline process. In order to avoid inconsistency, we generate 5 random sample sets and notify the mean classification accuracy obtained for entire dataset. Li et al. report a classification accuracy of 52.5% for SIFT with DoG [LP05] for the same dataset.

## 5. Conclusion

In this paper, we consider very high dimensional data which is not spatially correlated and accounts for large storage

space. Traditional techniques do not scale well with these kinds of datasets. Our attempt in the current case is to provide a framework for analyzing such datasets. We provide a visualization based interactive tool to guide the process of generating better clusters. We observed that distribution along 0 degree orientations is uniform and spread over a large range for several datasets. Though it takes considerable amount of time to analyze clusters, we can randomly sample a set of areas in the graph layout and observe the results which gives an overall view about the current process. We notice that though this framework has been designed for analyzing SIFT vectors from a given set of images, we can extend it to any high-dimensional dataset which requires a cluster analysis schema.

As noted in section 3, the number of joint-histograms that has to be pre-computed scales quadratically with the number of dimensions of the dataset. This makes the pre-processing time and used disk-space grow quadratically. We plan to include normalization method for PCP to further reduce the clutter. We also plan to provide an interactive interface where a user can re-order the dimensions displayed in PCP.

## References

- [03] <http://www-cvr.ai.uiuc.edu/ponce%20grp/data/index.html>.
- [05] <http://www.vlfeat.org/>.
- [06] <http://www.developer.nvidia.com/cuda>.
- [07] <http://www.openscenegraph.org/projects/osg>.
- [AdOL04] ARTERO A. O., DE OLIVEIRA M. C. F., LEVKOWITZ H.: Uncovering clusters in crowded parallel coordinates visualizations. In *INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization* (2004), pp. 81–88.
- [BL97] BLUM A. L., LANGLEY P.: Selection of relevant features and examples in machine learning. *Artificial Intelligence* 97, 1-2 (1997), 245–271.
- [CGR00] CHEN C., GAGAUDAKIS G., ROSIN P.: Content-based image visualization. *Fourth International Conference on Information Visualisation 0* (2000), 13.
- [CT76] CHERITION D., TARIAN R. E.: Finding minimum spanning trees. *SIAM journal on computing* 5 (1976), 724–741.
- [DB00] DY J. G., BRODLEY C. E.: Visualization and interactive feature selection for unsupervised data. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (2000), pp. 360–364.
- [DT05] DALAL N., TRIGGS B.: Histograms of oriented gradients for human detection. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1* (2005), pp. 886–893.
- [FWR99] FUA Y.-H., WARD M. O., RUNDENSTEINER E. A.: Hierarchical parallel coordinates for exploration of large datasets. In *VIS '99: Proceedings of the conference on Visualization '99* (1999), IEEE Computer Society Press, pp. 43–50.
- [GHGH09] GODIYAL A., HOBEROCK J., GARLAND M., HART J. C.: Rapid multipole graph drawing on the gpu. *Graph Drawing: 16th International Symposium, GD 2008, Heraklion, Crete, Greece, September 21-24, 2008. Revised Papers* (2009), 90–101.
- [HBV02] HALKIDI M., BATISTAKIS Y., VAZIRGIANNIS M.: Cluster validity methods: part i, ii. *SIGMOD Rec.* 31 (2002).



- [HFH\*09] HALL M., FRANK E., HOLMES G., PFAHRINGER B., REUTEMANN P., WITTEN I. H.: The weka data mining software: an update. *SIGKDD Explor. Newsl.* 11, 1 (2009), 10–18.
- [HJ05] HACHUL S., JÜNGER M.: *Large-Graph Layout with the Fast Multipole Multilevel Method*. Tech. rep., Zentrum für Angewandte Informatik Köln, Lehrstuhl Jünger, December 2005.
- [HNRL05] HUANG J. Z., NG M. K., RONG H., LI Z.: Automated variable weighting in k-means type clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 5 (2005), 657–668.
- [ID90] INSELBERG A., DIMSDALE B.: Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *IEEE visualization Proceedings of the 1st conference on Visualization* (1990), pp. 361–378.
- [IKH08] IMO J., KLENK S., HEIDEMANN G.: Interactive feature visualization for image retrieval. In *ICPR* (2008), pp. 1–4.
- [JCJ05] JOHANSSON J., COOPER M., JERN M.: 3-dimensional display for clustered multi-relational parallel coordinates. In *IV '05: Proceedings of the Ninth International Conference on Information Visualisation* (2005), pp. 188–193.
- [JD88] JAIN A. K., DUBES R. C.: *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.
- [JLJC05] JOHANSSON J., LJUNG P., JERN M., COOPER M.: Revealing structure within clustered parallel coordinates displays. In *INFOVIS '05: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization* (2005), p. 17.
- [JMCT83] J. M. CHAMBERS W. S. CLEVELAND B. K., TUKEY P. A.: *Graphical Methods for Data Analysis*. Chapman and Hall, New York, 1983.
- [Jol02] JOLLIFFE I. T.: *Principal Component Analysis*. Springer, 2002.
- [Kei02] KEIM D. A.: Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics* 8, 1 (2002), 1–8.
- [KJ97] KOHAVI R., JOHN G. H.: Wrappers for feature subset selection. *Artificial Intelligence* 97, 1-2 (1997), 273–324.
- [Low99] LOWE D. G.: Object recognition from local scale-invariant features. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2* (1999), p. 1150.
- [Low01] LOWE D. G.: Local feature view clustering for 3d object recognition. *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01) I* (2001), 682–688.
- [LP05] LI F.-F., PERONA P.: A bayesian hierarchical model for learning natural scene categories. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2* (2005), IEEE Computer Society, pp. 524–531.
- [MBM08] MAJI S., BERG A., MALIK J.: Classification using intersection kernel support vector machines is efficient. pp. 1–8.
- [MKO\*08] MUIGG P., KEHRER J., OELTZE S., PIRINGER H., DOLEISCH H., PREIM B., HAUSER H.: A four-level focus+context approach to interactive visual analysis of temporal features in large scientific data. *Computer Graphics Forum* 27, 3 (2008), 775–782.
- [MS04] MIKOLAJCZYK K., SCHMID C.: Scale & affine invariant interest point detectors. *International Journal on Computer Vision* 60, 1 (2004), 63–86.
- [NH01] NAKAZATO M., HUANG T. S.: 3d mars: Immersive virtual reality for content-based image retrieval. *IEEE International Conference on Multimedia and Expo* (2001).
- [Pet03] PETROVSKIY M. I.: Outlier detection algorithms in data mining systems. *Program. Comput. Softw.* 29, 4 (2003), 228–237.
- [RH99] RUI Y., HUANG T. S.: A novel relevance feedback technique in image retrieval. In *MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 2)* (New York, NY, USA, 1999), ACM, pp. 67–70.
- [RT01] RUBNER Y., TOMASI C.: *Perceptual Metrics for Image Database Navigation*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.
- [SLZ01] SU Z., LI S., ZHANG H.: Extraction of feature subspaces for content-based retrieval using relevance feedback. In *MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia* (New York, NY, USA, 2001), ACM, pp. 98–106.
- [SS05] SEO J., SHNEIDERMAN B.: A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization* 4, 2 (2005), 96–113.
- [Stu03] STUETZLE W.: Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample. *J. Classification* 20, 1 (2003), 025–047.
- [SZ03] SIVIC J., ZISSERMAN A.: Video google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision* (Oct. 2003), vol. 2, pp. 1470–1477.
- [SZM00] SU Z., ZHANG H., MA S.: Using bayesian classifier in relevant feedback of image retrieval. *Tools with Artificial Intelligence, IEEE International Conference on 0* (2000), 0258.
- [TC01] TONG S., CHANG E.: Support vector machine active learning for image retrieval. In *MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia* (New York, NY, USA, 2001), ACM, pp. 107–118.
- [vA02] ŠALTENIS V., AUŠRAITE J.: Data visualization: ideas, methods, and problems. *Informatics in education* 1, 1 (2002), 129–148.