Incorporating Shape into Histograms for CBIR

George Gagaudakis and Paul L. Rosin
Department of Information Systems and Computing
Brunel University
Uxbridge
Middlesex
UB8 3PH
UK
E-mail: {George.Gagaudakis, Paul.Rosin}@brunel.ac.uk

Abstract

This paper describes an indexing system for use in Content Based Image Retrieval. The standard colour histogram approach is simple, efficient, and robust. However, it does not include shape information, which leads to problems (e.g. many-to-many mappings). To remedy this we use additional features in an attempt to incorporate shape and textual information to the index key. Our experiments showed that the combination of colour, texture, distance and orientation histograms gave approximately 10% improvement of recall over the standard colour histogram.

1 Introduction

Computer technology faced a tremendous growth during the last decade, both in terms of computing speed and storage capabilities. Nowadays many people have access to large amounts of electronic data, much of which consists of images. Their storage has led to enormous image databases that are intrinsically harder to access and search than their textual counterparts. Explicitly identifying and entering descriptive keywords for each image by hand is impractical due to the overhead involved. Moreover, given the varied types of queries possible it is not usually possible to generate a complete set of keywords for each image. This implies that automated Content-Based Image Retrieval (CBIR) systems need to be developed.

Since Swain and Ballard's seminal paper [20] there has been considerable research carried out in the area of CBIR [3, 8]. Given the enormous difficulties in reliably identifying objects in images the majority of work has been constrained to perform retrieval at a fairly primitive level. Rather than search for images based on their semantic content (e.g. "find all pictures of young, smiling, faces") automated indexing of the database images is usually based on colour or geometric features extracted from the image. The indexing process takes place in both the phases of creating/updating and searching the database, as shown in figure 1.

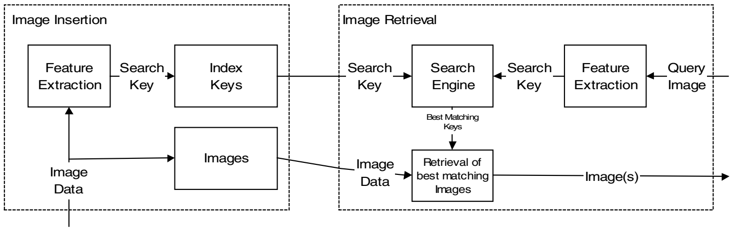


Figure 1: Overall view of a CBIR system

Swain & Ballard matched images based solely on their colour. The distribution of colour was represented by colour histograms, and formed the images' feature vectors. The similarity between a pair of images was then calculated using a similarity measure between their histograms called the "normalised histogram intersection". This approach became very popular due to its advantages:

- *Robustness.* The colour histogram is invariant to rotation of the image on the view axis, and changes in small steps when rotated otherwise or scaled [20]. It is also insensitive to changes in image and histogram resolution and occlusion.
- *Effectiveness.* There is high percentage of relevance between the query image and the extracted matching images.
- *Implementation simplicity.* The construction of the colour histogram is a simple scanning of the image, to get the colour values, discretisation of the colour values to the resolution of the histogram, and building the histogram using colour components as indices.
- *Computational simplicity.* The histogram computation has $O(M^2)$ complexity for images of size $M \times M$. The complexity for a single image match is linear, $O(n)$, where n represents the number of different colours, or resolution of the histogram.
- *Low storage requirements.* The colour histogram size is significantly smaller than the image itself, assuming colour quantisation.

There do remain some problems with the colour histogram though, namely:

- Different images may have similar or identical colour histograms. For instance, as an extreme case, randomly scrambling the positions of pixels in an image leaves its histogram unaffected despite massive changes in the image content.
- Images taken under different ambient lighting may produce different histograms. This has been partly addressed by applying colour constancy normalisation [4].

In the intervening years many more sophisticated methods have been developed. Attempts have been made to identify objects (e.g. people, horses, trees) to drive the matching process [5]. However, this is extremely difficult, requiring large systems that have specialized algorithms for identifying each type of object (e.g. tree identifiers with sub-algorithms necessary for different types of trees). Thus with the current state of the art such an approach is not general purpose or easily extensible without significant scaling problems.

More practical approaches are still rooted in low level feature extraction and description. Shape is potentially an extremely useful and powerful feature, but shape-based image systems generally run into two problems. First, they mostly require the image to be partitioned into regions from which shape descriptors can then be extracted. Unless the segmentation is directed by the user (e.g. [14]) segmentation algorithms are prone to fail, especially when new situations due to different imaging modality or object type are present [6]. Second, determining effective shape descriptors for complex natural objects is still an active area of research [12].

Given the inherent difficulties of methods requiring segmentation some authors have built in a spatial component into CBIR by splitting the image up based on a fixed grid or non-regular cells (e.g. Stricker and Dimai's fuzzy oval [19]). Standard analysis techniques (e.g. colour histograms) can then be performed in each cells. However, this approach is still static and does not adapt to the image content. To this effect our paper revisits and extends the traditional colour histogram based approach in a more dynamic manner. Our approach is to capitalise on the benefits of histogram matching and to overcome its limitations by incorporating some aspects of shape into the matching process.

2 Techniques for Incorporating Shape

Since both extracting shapes and describing them is problematic several methods for indirectly incorporating shape will be considered. Whereas it is considered extremely difficult to perform semantically meaningful segmentation many reasonably reliable algorithms for low-level feature extraction have been developed. These will be used to provide the spatial information that is lacking in colour histograms. Rather than attempt to directly measure shape we will calculate some simpler properties that are indirectly related to shape and avoid the requirement for good segmentation, providing a more practical solution.

Previous work in this vein is given by Jain and Vailaya [9] who in combination with colour histograms they use edge orientation histograms, which encode some aspects of shape information, enabling querying to be more responsive to the shape content of the images. Moreover to extract the shape information only standard edge detection is required (e.g. Canny's algorithm [2]), and minor errors in the edge map have little effect on the edge orientation histograms. Unlike colour histograms the orientation histograms are not rotationally invariant, and so the histogram matching process has to iteratively shift the histogram to find the best correspondence. A more important consideration is that the edge maps were thresholded by some unspecified means. For robustness an adaptive thresholding scheme should be used [16]. However, an alternative is to entirely dispense with thresholding and include all edges, weighting their contribution to the histogram by their magnitudes so as to reduce the contribution from spurious edges. This is the approach we take in the reported experiments.

2.1 Multi-resolution Salience Distance Transform

Another approach to including shape information is based on the distance transform (DT). The DT is a method for taking a binary image of feature and non-feature pixels and calculating at every pixel in the image the distance to the closest feature. Although this is a potentially expensive operation efficient algorithms have been developed that only require two passes through the image [1]. Just as a limitation of Jain and Vailaya's approach was its application of thresholding, this also causes problems for distance transforms. To improve the stability of the distance transform Rosin and West [17] developed a weighted version called the salience distance transform (SDT). Rather than propagating out Euclidean (or quasi- Euclidean) distances from edges the distances are weighted by the salience of the edge. Various forms of salience were demonstrated, incorporating features such as edge magnitude, curve length, and local curvature. The effect of including salience was to downplay the effect of spurious edges by soft assignment while avoiding the sensitivity problems of thresholding.

To further improve performance the edge detection was applied over a range of scales. Rosin and West performed the SDT on the edges at each scale and then combined them. Instead we first combine the gradient maps over the scales (eight scales are used in the experiments), apply maximal suppression produce a single edge map which is then used to generate the distance map. The advantage of this modification is that similar results are obtained with considerably less computation.

Once the SDT has been performed the distance values are histogrammed. It can be seen that the histograms will respond differently to different types of shapes. First there is the crude distinction between cluttered, complex scenes and simple sparse scenes, which will result in different ends of the histogram being heavily populated. In that respect the distance histograms area provide an indication of image complexity, along the lines of Kawaguchi and Taniguchi's [11]. However, rather than return a single complexity measurement the shape of the histogram will indicate more subtle distinctions between shapes.

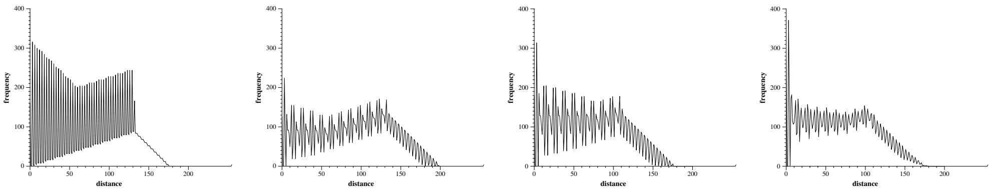


Figure 2: Distance histograms for simple geometric shapes

As a simple example figure 2 shows distance histograms calculated for 128×128 images each containing one geometric shape: a square, its inscribed circle, its circumscribed circle, and a noisy version of the circumscribed circle. The jaggy nature of the graphs is due to the effects of spatial discretisation of the image. It can be seen that the square and regular circle histograms are somewhat different in appearance. Also, the addition of noise is reflected by increased bin counts at the lower end of the noisy circle histogram.

2.2 Segmentation by Thesholding

The next approach to incorporating shape information is based on a form of segmentation. We previously stated that reliable segmentation is currently not possible. Therefore we do not attempt to perform true segmentation into meaningful regions. Instead the simplest possible sort of segmentation is carried out, namely binary thresholding. While binarisation can produce meaningful regions for well-defined bimodal images it is unlikely to do so for less constrained, more naturalistic images. However, this is not our goal. All that is required is that a reliable spatial partitioning of the image is provided; the regions formed may be purely arbitrary in terms of their correspondence with objects in the scene.

The segmentation performs the same function as the partitioning based approaches by Dimai and Stricker and others. The partitioning injects the spatial information into the analysis so that standard feature based (e.g. non-spatial) methods can then be applied within each region. In fact we only consider the two classes black and white each as single composite regions rather than treat each individual region separately. Not only does this avoid the need for component labelling but it also reduces sensitivity to variations in the thresholding process. The advantage of using of thresholding to provide the image partitioning is that it is adaptive to the image content unlike the rigid grid methods used by others.

There has been considerable work carried out on thresholding methods [18] to ensure that the method used was reliable. Tests were carried out to compare several thresholding algorithms. Three representative types of algorithm were used, based on entropy [10], statistics [15] and moments [21]. Several images were used, and from each one several random windows were extracted. The whole images and the selected windows were thresholded, and the results differenced to determine errors, i.e. the percentage of thresholded pixels that differed. We found that the average errors were not of significant difference to classify a "best" method; we used Kapur's method for all the future experiments.

2.3 Texture

A common extension to colour CBIR systems is to add textural information. There are many texture analysis methods available, and these can be applied either to perform segmentation of the image, or to extract texture properties from segmented regions or the whole image. In keeping with the histogram approach, we used He and Wang's approach [7], which generates a histogram, called the texture spectrum.

The first step is to analyse each pixel within its 3×3 neighbourhood. A vector $V = \{V_1, \ldots, V_8\}$ is constructed where V_c represents the intensity of the central pixel and V_i are the intensities of its neighbours. V is then transformed to a *texture unit* $TU = \{E_1, \ldots, E_8\}$. Each texture is then mapped onto a unique integer forming the *texture unit number* N_{TU}.

$$E_i = \begin{cases} 0 & \text{if } V_i < V_c \\ 1 & \text{if } V_i = V_c \\ 2 & \text{if } V_i > V_c \end{cases} .; \; N_{TU} = \sum_{i=1}^{8} 3^{i-1} E_i .$$

The frequencies of texture unit numbers are accumulated to form the texture spectrum, which we then treat as a texture histogram. To improve efficiency and memory requirements we have modified the texture unit and number to be

$$E_i = \begin{cases} 0 & \text{if } V_i < V_c \\ 1 & \text{if } V_i \geq V_c \end{cases} ; \; N_{TU} = \sum_{i=1}^{8} 2^{i-1} E_i .$$

This reduces the number of unique texture unit numbers from 6561 to 256 with little loss in useful discriminative power.

To summarise this section, we are attempting to incorporate shape, in addition to the colour histogram, by means of, edge orientation and edge distances. Additionally texture will add more spatial information.

2.4 Combining Similarity Measures

Having generated histograms based on the different properties (colour, texture, and shape) the histograms of the query images are compared against the corresponding histograms of the database images using normalised histogram intersection. There remains the issue of how the similarity measures from each property are combined to achieve a single similarity rating so that the database images can be easily ranked according to their closeness with the query image.

One approach is to use a weighted sum, as in [9]

$$S_t = \frac{w_1 S_1 + w_2 S_2 + \ldots + w_n S_n +}{w_1 + w_2 + \ldots + w_n}$$

where, S_t is the similarity of two images combined over the similarity indices, S_i is the ith type of similarity index between the two images and w_i is the weighting factor for each index. Its drawback is that it requires careful selection of the weight values to obtain good results. In general, colour-based queries are more accurate than shape based ones, so one would use higher weight value for the colour similarity [9]. However, choosing appropriate weights will be dependent on the contents of the database, and generally requires extensive experimentation.

The above difficulties arise because the different similarity measures are incommensurate. We believe a better approach is to use the geometric mean rather than the weighted sum

$$S_t = \sqrt[n]{S_1 S_2 \ldots S_n}$$

which does not require any weighting factors.

3 Experimental Results

Our objective is to improve the effectiveness of the simple colour histogram by incorporating shape, using simple methods, as an alternative to precise segmentation. At this point we examine the performance of these potential methods. There are two main aspects of interest in this context, namely, *effectiveness* and *efficiency*.

Efficiency is a measure closely related with the storage requirements and responsiveness of a CBIR system. At this point of our research we are not concerned with all the aspects of efficiency, such as the histogram size, apart from retaining to some degree the simplicity of the simple colour histogram. Modifications of histogram sizes are possible and able to increase efficiency at certain trade-offs, depending on the case.

Effectiveness is a measure of the relevance of retrieved images to a query, as perceived by the user. Trying to address this, we employed two values, namely, *recall* and *precision* Recall represents the proportion of 'correct' matches in the top-ten list of the retrieved images. Precision represents the number of 'correct' images that are retrieved among the top-ten hits. Alternatively we represent precision by the distance between the first and the last 'correct' retrieved image; the smaller the value the better the precision is.

Using the above measures, an experimental data-set, for which the ground-truth classification is available, is essential. It is not hard to create a collection of still images that can be clustered into disjoint classes, as long as the content is simple (such as faces, buildings, landscapes etc.). However it becomes harder as the number of classes increases and distinctions become more subtle to complete the collection without the evaluation becoming very subjective.

To avoid the difficulty (and impracticality) of manual groundtruthing we followed a method similar to Milanese *et al.* [13] where still images extracted from video clips are used. A broadcast TV signal from a local Greek station (CRETA Channel) was captured at a resolution of two frames-per-second. This was then resampled to obtain 9-11 still images representing each clip. As in [13] we assumed that (a) the continuity of the visual content of a clip is implied by the uninterrupted recording of a video camera, (b) there is gradual change of content, from frame to frame, due to camera operations and subject motion, object appearance and disappearance.

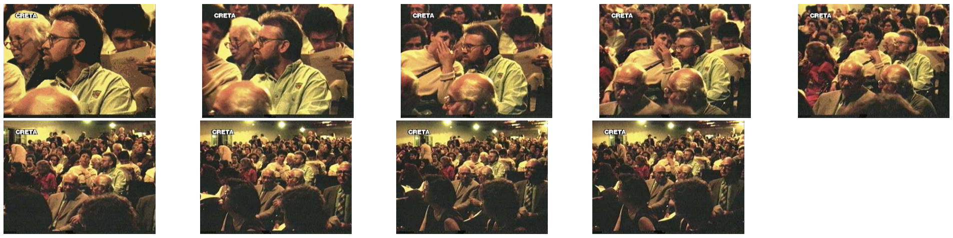


Figure 3: Example of images extracted from a clip

For the evaluation of the potential methods (histogram combinations), we created 39 queries. The query images were randomly selected from the data set, so all the video clips are represented.

By combining all the techniques mentioned above we came up with 48 different types of indexing. We run the queries and from the responses we calculated the effectiveness measures for each one. In table 2, the measure averages of some methods are illustrated, sorted by computational complexity, while in table 1 the acronym of each histogram is described.

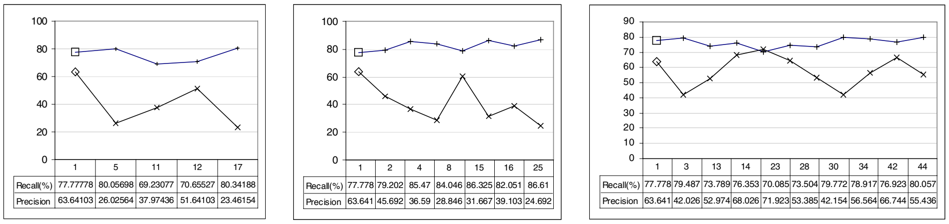
Our analysis of the results is focused on the effect of combining different histograms, against the simple colour histogram (SCH). We have, roughly, classified the indexing methods of table 2, into four disjoint classes.

Texture and/or Shape (TS). This type of indexing combines the TS feature histograms only, without the colour histogram. The results showed that, some TS combinations (5 and 17) outperform the SCH, in both recall and precision. In figure 4*a*, this is illustrated graphically.

Colour, Texture and/or Shape (CTS). The histogram combinations are extended to include colour. Generally, the performance is again better, as expected, than the SCH. Taking the averages of the methods (TS and CTS) show that CTS indexing has better average recall but slightly lower precision. The best performing combination is 25.

Including Spatial Partitioning (ISP). As mentioned, we used a thresholding algorithm to perform a partitioning of the images. In this type of indexing we combine the texture and colour histograms of the two image partitions, with the orientation and distance histogram. The performance in average is improved again, in some cases, against the SCH, while the best combination 44, performs worse in all terms than method 25. A graphical illustration of the ISP performance is shown in figure 4*c*.

Colour, Texture, Shape and/or Partitioning (CTSP). In this category we use the Colour and Texture histograms of the whole image as well as the histograms included in ISP. As figure 5 shows, on average it is close to the SCH , but rarely improves it, and is outperformed by combination 25 (CTS).

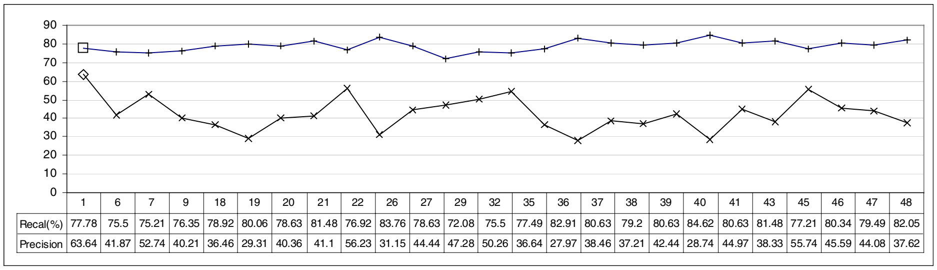


	1	5	11	12	17
Recall(%)	77.77778	80.05698	69.23077	70.65527	80.34188
Precision	63.64103	26.02564	37.97436	51.64103	23.46154

	1	2	4	8	15	16	25
Recall(%)	77.778	79.202	85.47	84.046	86.325	82.051	86.61
Precision	63.641	45.692	36.59	28.846	31.667	39.103	24.692

	1	3	13	14	23	28	30	34	42	44
Recall(%)	77.778	79.487	73.789	76.353	70.085	73.504	79.772	78.917	76.923	80.057
Precision	63.641	42.026	52.974	68.026	71.923	53.385	42.154	56.564	66.744	55.436

Figure 4: Performance graphs of (a) TS, (b) CTS and (c) ISP indexing



	1	6	7	9	18	19	20	21	22	26	27	29	32	35	36	37	38	39	40	41	43	45	46	47	48
Recal(%)	77.78	75.5	75.21	76.35	78.92	80.06	78.63	81.48	76.92	83.76	78.63	72.08	75.5	77.49	82.91	80.63	79.2	80.63	84.62	80.63	81.48	77.21	80.34	79.49	82.05
Precision	63.64	41.87	52.74	40.21	36.46	29.31	40.36	41.1	56.23	31.15	44.44	47.28	50.26	36.64	27.97	38.46	37.21	42.44	28.74	44.97	38.33	55.74	45.59	44.08	37.62

Figure 5: Performance graph of CTSP indexing

4 Conclusions

A number of different histogram combinations have been presented in this paper, offering an improved means of image indexing to the traditional colour histogram. A multi-resolution salience distance transform and an edge magnitude weighted orientation histograms were used as a means to incorporate shape along with textual information. Additionally, we used Kapur's entropy thresholding as a way to include some spatial information that could then be employed by standard indexing schemes (e.g., colour histogramming).

In many cases the spatial partitioning improved indexing so as to outperform the simple colour histogram. However, the best performance in terms of recall and precision was achieved by combining colour, texture, distance and orientation alone without spatial partitioning. Future work will concentrate on verifying these results by extending the evaluation to test larger datasets. Since most of the groundtruthing is automated the test methodology scales well.

References

[1] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics and Image Processing*, 34(3):344–371, 1986.

[2] J. Canny. A computational approach to edge detection. *IEEE Trans. PAMI*, 8:679–698, 1986.

[3] M. Demarsicoi, L. Cinque, and S. Levialdi. Indexing pictorial documents by their content - a survey of current techniques. *Image and Vision Computing*, 15(2):119–141, 1997.

[4] M.S. Drew, J. Wei, and Z. Li. Illumination-invariant color object recognition via compressed chromaticity histograms of normalized images. Technical report, Simon Fraser University, Canada, 1997.

[5] D.A. Forsyth, J. Malik, M.M. Fleck, T. Leung, C. Bregler, C. Carson, and H. Greenspan. Finding pictures of objects in large collections of images. In *Proceedings of International Workshop on Object Recognition*, 1996.

[6] E.M. Gurari and H. Wechsler. On the difficulties involved in the segmentation of pictures. *IEEE Trans. PAMI*, 4(3):304–306, 1982.

[7] D.C. He and L. Wang. Texture unit, texture spectrum, and texture analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4), 1990.

[8] F. Idris and S. Panchanathan. Review of image and video indexing techniques. *Journal of Visual Communication and Representation*, 8(2):149–166, 1997.

[9] A.K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233–1244, 1996.

[10] J.N. Kapur, P.K. Sahoo, and A.K.C. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics and Image Processing*, 29(3):273–285, 1985.

[11] E. Kawaguchi and R.I. Taniguchi. The depth first [icture-expression as an image thresholding strategy. *IEEE Trans. on Sys., Man, and Cybernetics*, 19(5):1321–1329, 1989.

[12] K.V. Mardia. *Statistics of Directional Data* Academic Press, 1972.

[13] R. Milanese and M. Cherbuliez. A rotation, translation, and scale-invariant approach to content-based image retrieval. *Journal of Visual Communication and Image Representation*, 10:186–196, 1999.

[14] E.N. Mortensen and W.A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60(5):349–384, 1998.

[15] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Sys., Man, and Cybernetics*, 9:62–66, 1979.

[16] P.L. Rosin. Edges: saliency measures and automatic thresholding. *Machine Vision and Applications*, 9(4):139–159, 1997.

[17] P.L. Rosin and G.A.W. West. Salience distance transforms. *Graphical Models and Image Processing*, 57:483–521, 1995.

[18] P.K. Sahoo, S. Soltani, A.K.C. Wong, and Y.C. Chen. A survey of thresholding techniques. *Computer Vision, Graphics and Image Processing*, 41(2):233–260, 1988.

[19] M. Stricker and A. Dimai. Spectral covariance and fuzzy regions for image indexing. *Machine Vision and Applications*, 10(2):66–73, 1997.

[20] M.J. Swain and D.H. Ballard. Color indexing. *Int. J. Computer Vision*, 7(1):11–32, 1991.

[21] W.H. Tsai. Moment-preserving thresholding. *Computer Vision, Graphics and Image Processing*, 29:377–393, 1985.

Label	**Method**
cd	Simple colour histogram
cd0 / cd1	Colour histograms of the masked and unmasked areas
Td	Simple texture histogram
td0 / td1	Texture histogram of the masked and unmasked areas
Dd	Histogram of the multi-scale salience distance map
Dird	Edge magnitude weighted orientation histogram

Table 1: Associations between labels and histogram.

Label	**Methods Used**	**Recall (%)**	**Precision**
1	cd	77.77778	63.64103
2	cd + td	79.20228	45.69231
3	cd0 + cd1 + dird	79.48718	42.02564
4	cd + dird	85.47009	36.58974
5	td + dird	80.05698	26.02564
6	cd0 + cd1 + td	75.49858	41.87179
7	cd + td0 + td1	75.21368	52.74359
8	cd + td + dird	84.04558	28.84615
10	cd + dd	78.91738	60.20513
14	cd0 + cd1 + td0 + td1	76.35328	68.02564
15	cd + dd + dird	86.32479	31.66667
16	cd + td + dd	82.05128	39.10256
17	td + dd + dird	80.34188	23.46154
18	td + td0 + td1 + dird	78.91738	36.46154
19	cd0 + cd1 + td + dird	80.05698	29.30769
20	cd + cd0 + cd1 + dird	78.63248	40.35897
21	cd + td0 + td1 + dird	81.48148	41.10256
22	cd + cd0 + cd1 + td0 + td1	76.92308	56.23077
25	**cd + td + dd + dird**	**86.60969**	**24.69231**
26	cd + cd0 + cd1 + td + dird	83.76068	31.15385
27	cd + cd0 + cd1 + td + td0 + td1	78.63248	44.43590
30	td0 + cd1 + dd + dird	79.77208	42.15385
31	cd0 + cd1 + td + dd	78.91738	38.53846
34	cd0 + cd1 + td0 + td1 + dird	78.91738	56.56410
35	td + td0 + td1 + dd + dird	77.49288	36.64103
36	cd0 + cd1 + td + dd + dird	82.90598	27.97436
37	cd + cd0 + cd1 + dd + dird	80.62678	38.46154
38	cd + cd0 + cd1 + td + dd	79.20228	37.20513
39	d + td0 + td1 + dd + dird	80.62678	42.43590
40	cd + cd0 + cd1 + td + dd + dird	84.61539	28.74359
41	cd + cd0 + cd1 + td0 + td1 + dird	80.62678	44.97436
42	cd0 + cd1 + td0 + td1 + dd	76.92308	66.74359
43	cd + cd0 + cd1 + td + td0 + td1 + dird	81.48148	38.33333
44	cd0 + cd1 + td0 + td1 + dd + dird	80.05698	55.43590
45	cd + cd0 + cd1 + td0 + td1 + dd	77.20798	55.74359
46	cd + cd0 + cd1 + td0 + td1 + dd + dird	80.34188	45.58974
47	cd + cd0 + cd1 + td + td0 + td1 + dd	79.48718	44.07692
48	cd + cd0 + cd1 + td + td0 + td1 + dd + dird	82.05128	37.61538

Table 2: Performance of histogram combinations, sorted by computational and storage requirements.