An Efficient Distance Between Multi-dimensional Histograms for Comparing Images

Francesc Serratosa and Gerard Sanromà

Universitat Rovira i Virgili, Dept. d'Enginyeria Informàtica i Matemàtiques, Spain
francesc.serratosa@.urv.cat, gerard.sanroma@urv.cat

Abstract. The aim of this paper is to present an efficient distance between n-dimensional histograms. Some image classification or image retrieval techniques use the distance between histograms as a first step of the classification process. For this reason, some algorithms that find the distance between histograms have been proposed in the literature. Nevertheless, most of this research has been applied on one-dimensional histograms due to the computation of a distance between multi-dimensional histograms is very expensive. In this paper, we present an efficient method to compare multi-dimensional histograms in O(2z), where z represents the number of bins. Results show a huge reduction of the time consuming with no recognition-ratio reduction.

1 Introduction

Finding the distance or similarity between histograms is an important issue in image classification or image retrieval since a histogram represents the frequency of the values of the pixels among the images. For this reason, a number of measures of similarity between histograms have been proposed and used in computer vision and pattern recognition. Moreover, if the position of the pixels is unimportant while considering the distance measure, we can compute the distance between images in an efficient way by computing the distance between their histograms.

Most of the distance measures presented in the literature (there is an interesting compilation in [1]) consider the overlap or intersection between two histograms as a function of the distance value but they do not take into account the similarity on the non-overlapping parts of the two histograms. For this reason, Rubner presented in [2] a new definition of the distance measure between n-dimensional histograms that overcomes this non-overlapping parts problem. It was called Earth Mover's Distance and it is defined as the minimum amount of work that must be performed to transform one histogram into the other one by moving distribution mass.

Often, for specific set measurements, only a small fraction of the *bins* in a histogram contain significant information, that is, most of the *bins* are empty. This is more frequent when the dimensions of the histograms increase. In that cases, the methods that use histograms as fixed-sized structures obtain poor efficiency. In the algorithm depicted by Rubner [2] to find the Earth Mover's Distance the empty-bins where not explicitly considered. They used the simplex algorithm [3] to compute the distance measure and the method presented in [4] to search a good initialisation. The computational cost of the simplex iteration is $O(z'^2)$, where z' is the number of

D.-Y. Yeung et al. (Eds.): SSPR&SPR 2006, LNCS 4109, pp. 412 – 421, 2006.

non-empty bins. The main drawback of this method is that the number of iterations is not bounded. Moreover, the initialisation cost is *O(2z')*.

To reduce the computational cost, Cha presented in [1] three algorithms to obtain the Earth Mover's Distance between one-dimensional histograms when the type of measurements where *nominal*, *ordinal* and *modulo* in *O(z)*, *O(z)* and $O(z^2)$ respectively, being *z* the number of levels or bins.

Finally, Serratosa reduced more the computational cost in [5]. They presented three new algorithms to compute the Earth Mover's Distance between one-dimensional histograms when the type of measurements where *nominal*, *ordinal* and *modulo*. The computational cost were reduced to *O(z')*, *O(z')* and $O(z'^2)$ respectively, being *z'* the number of non-empty bins.

It was presented in [6] an algorithm to compute the distance between histograms that the input was a built histogram (obtained from the target image) and another image. Then, it was not necessary to build the histogram of the image of the database to compute the distance between histograms.

Really few have been done to compare n-dimensional histograms except in [2]. The main drawback of the method presented in [2] is the computational cost. In this paper, we present an efficient algorithm to compute the distance between n-dimensional histograms with a computational cost of *O(2z)*. Our algorithm does not depend on the type of measurements (*nominal*, *ordinal* or *modulo*). In the next section, we define the histograms and types of values. In section 3, we give the definitions of distances between histograms and between sets and in section 4 we show the algorithm to compute the distance between histograms. In sections 5 and 6 we show the experimental validation of our algorithm and the conclusions.

2 Sets and Histograms

In this section, we formally give a definition of histograms. Moreover, we show a property obtained from the definition of the histograms that will be useful in the definitions of the distances given in the next section. Finally, we define the distance between pixel values.

2.1 Histogram Definition

Let *x* be a measurement which can have one of *z* values contained in the set $X=\{x_1,...x_z\}$. Each value can be represented in a *T*-dimensional vector as $x_i=(x_i^1, x_i^2,...,x_i^T)$. Consider a set of *n* elements whose measurements of the value of *x* are $A=\{a_1,...a_n\}$ where $a_t \in X$ being $a_t=(a_t^1, a_t^2,...,a_t^T)$.

The histogram of the set *A* along measurement *x* is *H(x,A)* which is an ordered list consisting of the number of occurrences of the discrete values of *x* among the a_t. As we are interested only in comparing the histograms and sets of the same measurement *x*, *H(A)* will be used instead of *H(x,A)* without loss of generality. If $H_i(A)$, $1 \le i \le z$, denotes the number of elements of *A* that have value x_i, then $H(A)=[H_1(A), ...,H_z(A)]$ where

$$H_i(A)=\sum_{t=1}^{n} C_{it}^{A} \qquad (1)$$

and the individual costs are defined as

$$C_{i,t}^{A} = \begin{cases} 1 & if\ a_t = x_i \\ 0 & otherwise \end{cases} \tag{2}$$

The elements $H_i(A)$ are usually called *bins* of the histogram. Note that z is the number of bins of the histogram. In a T-dimensional histogram with m values per each dimension, the number of bins is $z=m^T$.

2.2 Property of the Individual Costs

Given a value a_t, the addition of all the individual costs is 1.

$$\sum_{i=1}^{z} C_{i,t}^{A} = 1 \quad 1 \le t \le n \tag{3}$$

Proof

Given the t-element of the set A, this element has only one value a_t. Therefore, there is only one value of i such that $C_{i,t}^{A} = 1$ (when $a_t = x_i$) and for all the other values of i, $C_{i,t}^{A} = 0$ (that is, $a_t \neq x_i$). Then, the addition of all the values is one.

2.3 Type of Measurements and Distance Between Them

The distance between histograms presented in this paper is used as a fast method for comparing images and image retrieval. The most used colour representations are base on the R,G,B or H,S,I descriptors. The hue parameter (H) is a modulo-type measurement (measurement values are ordered but form a ring due to the arithmetic modulo operation) and the other parameters are ordinal-type measurements.

Corresponding to these types of measurements mentioned before, we define a measure of difference between two measurement levels $a=(a^1, a^2,\dots,a^T) \in X$ and $b=(b^1, b^2,\dots,b^T) \in X$ as follows:

$$d(a,b) = \sum_{j=1}^{T} S \text{ where } S = \begin{cases} m - \left|a^j - b^j\right| & if\ a^j - b^j \le m/2\ and\ a^j, b^j \in Modulo\ type \\ \left|a^j - b^j\right| & otherwise \end{cases} \tag{4}$$

This measure satisfy the following necessary properties of a metric. Since they are straightforward facts, we omit the proofs. The proof of the triangle inequality for the modulo distance is depicted in [1] for the one-dimensional case ($T=1$).

3 Distance Definitions

In this section we present the distance between sets $D(A,B)$ and the distance between their histograms $D(H(A),H(B))$. We proof that both satisfy the necessary properties of a metric and that the distance values are the same, $D(A,B) = D(H(A),H(B))$. To do so, we find a relation between the assignments between elements of the sets A and B while computing $D(A,B)$ and the assignments between *bins* while computing $D(H(A),H(B))$.

This is an important result since the computational cost of *D(A,B)* is exponential respect the number of the set elements, *n*, but the computational cost of *D(H(A),H(B))* is only quadratic respect the number of *bins* of the histogram *z*. Moreover, in most of the applications, *z* is much smaller than *n*. Another advantage is that the time consuming of the comparison is constant and does not depend on each set.

3.1 Distance Between Sets

Given two sets of *n* elements, *A* and *B*, the distance measure is considered as the problem of finding the minimum difference of pair assignments between both sets. That is, to determine the best one-to-one assignment *f* (bijective function) between the sets such that the sum of all the differences between two individual elements in a pair $a_i \in A$ and $b_{f(i)} \in B$ is minimised.

$$D(A,B) = \min_{\forall f A \to B} \left(\sum_{t=1}^{n} d\left(a_t, b_{f(t)}\right) \right) \tag{5}$$

We are interested only in the *D(A,B)* value rather than the assignment *f*. Nevertheless, we call f_{opt} as the assignment such that the distance is obtained, so we can redefine the distance as follows,

$$D(A,B) = \sum_{t=1}^{n} d\left(a_t, b_{f_{opt}(t)}\right) \tag{6}$$

3.2 Distance Between Histograms

The distance between histograms that we present here is a generalisation of the Earth Mover's Distance presented in [2]. Intuitively, given two T-dimensional histograms, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the distance measure is the least amount of work needed to fill the holes with earth. Here, a unit of work corresponds to transporting a unit of earth by a unit of ground distance.

More formally, given two histograms *H(A)* and *H(B)*, where measurements can have one of *z* values contained in the set $X=\{x_1,...x_z\}$, the distance between the histograms *D(H(A),H(B))* is defined as follows,

$$D(H(A),H(B)) = \min_{\forall f A \to B} \left(\sum_{i,j=1}^{z} d\left(x_i, x_j\right) g_f(i,j) \right) \tag{7}$$

The flow between the *bins* of both histograms is represented by $g_f(i,j)$, that is, the mass of earth that is moved as one unit from the *bin i* to the *bin j*. The product $d(x_i,x_j)g_f(i,j)$ represents the work needed to transport this mass of earth. Similarly to equation (5), we can redefine the distance using the optimal assignment f_{opt},

$$D(H(A),H(B)) = \sum_{i,j=1}^{z} d\left(x_i, x_j\right) g_{f_{opt}}(i,j) \tag{8}$$

3.2.1 New Definition of the Flow Between Bins

In the definition of the distance between histograms presented in [2], the flow between histograms was shown to be a bi-dimensional matrix. The rows of the matrix represented the *bins* of one of the histograms and the columns represented the *bins* of the other histogram. Thus, each value of a matrix element was the flow between both *bins*. In that paper, there was no relation between the distance between the sets, *D(A,B)*, and the distance between the histograms of these sets, *D(H(A),H(B))*. For this reason, in the definition of the flow between *bins*, some constraints were needed to be imposed to match the distance definition to the transportation problem.

In our paper, we determine the flow between *bins* $g_f(i,j)$, as a function of the one-to-one assignment f between the sets A and B used to compute the distance *D(A,B)* as follows,

$$g_f(i,j)=\sum_{t=1}^{n} C_{i,t}^{A} C_{j,f(t)}^{B} \qquad 1\le i,j\le z \tag{9}$$

were the costs C are given in (2).

With this new definition, we obtain two advantages; First, there is a relation between distances *D(A,B)* and *D(H(A),H(B))* through their definition. Second, the constraints arbitrarily imposed to the flow between *bins* in [2], were converted in deducted properties that make possible to naturally match the distance between histograms to the transportation problem.

3.2.2 Properties of the Flow $g_f(i,j)$

The flow between the *bin* i of the set A and the *bin* j of the set B through the assignment f fulfils the following three properties,

Property a) $g_f(i,j)\ge 0 \quad 1\le i,j\le z$

Property b) $\sum_{j=1}^{z} g_f(i,j)=H_i(A) \quad 1\le i\le z$

Property c) $\sum_{i=1}^{z} g_f(i,j)=H_j(B) \quad 1\le j\le z$

Proofs

Property (a) is a straightforward fact due to equations (2) and (9).

Property (b) Using equation (9), we obtain that $\sum_{j=1}^{z} g_f(i,j)=\sum_{j=1}^{z}\sum_{t=1}^{n} C_{i,t}^{A} C_{j,f(t)}^{B}$, and exchanging the sumatories, we obtain that $\sum_{j=1}^{z} g_f(i,j)=\sum_{t=1}^{n}\sum_{j=1}^{z} C_{i,t}^{A} C_{j,f(t)}^{B}$. Then, if we spawn the external sumatory, we have the following formula, $C_{i,1}^{A}\sum_{j=1}^{z} C_{j,f(1)}^{B}+C_{i,2}^{A}\sum_{j=1}^{z} C_{j,f(2)}^{B}+\ldots+C_{i,n}^{A}\sum_{j=1}^{z} C_{j,f(n)}^{B}$ that can be reduced to $C_{i,1}^{A}+C_{i,2}^{A}+C_{i,3}^{A}+\ldots+C_{i,n}^{A}$ do to equation (3) and considering that f is bijective. So, we arrive at the expression $\sum_{j=1}^{z} g_f(i,j)=\sum_{t=1}^{n} C_{it}^{A}=H_i(A)\cdot$

Property (c) Using equation (9), we obtain that $\sum_{i=1}^{z} g_f(i,j) = \sum_{i=1}^{z}\sum_{t=1}^{n} C_{i,t}^{A} C_{j,f(t)}^{B}$, and exchanging the sumatories and the order of the costs, we obtain that $\sum_{i=1}^{z} g_f(i,j) = \sum_{t=1}^{n}\sum_{i=1}^{z} C_{j,f(t)}^{B} C_{i,t}^{A}$. Then, if we spawn the external sumatory, we have the following formula, $C_{j,f(1)}^{B}\sum_{i=1}^{z} C_{i,1}^{A} + C_{j,f(2)}^{B}\sum_{i=1}^{z} C_{i,2}^{A} + \ldots + C_{j,f(n)}^{B}\sum_{i=1}^{z} C_{i,n}^{A}$. Finally, applying equation (3), this sumatory is reduced to $C_{j,f(1)}^{B} + C_{j,f(2)}^{B} + \ldots + C_{j,f(n)}^{B}$. And so, $\sum_{i=1}^{z} g_f(i,j) = \sum_{t=1}^{n} C_{j,f(t)}^{B} = H_j(B)$.

3.3 Properties of the Distances

We present in this section the metric properties of the distances between sets and histograms. Moreover, we show that the distance value of these distances is the same. To that aim, we first describe a lemma. We assume that there are two measurement sets *A* and *B* that have *n* elements contained in the set *X={x₁,...x_z}*.

Lemma

The distance between two elements of the sets *A* and *B* given an assignment *f*, can be obtained as the distance between *bins* as follows,

$$d\left(a_t, b_{f(t)}\right) = \sum_{i,j=1}^{z} C_{i,t}^{A} C_{j,f(t)}^{B} d\left(x_i, x_j\right) \quad 1 \le t \le n \quad f \text{ bijective} \tag{10}$$

Proof

By definition of the individual cost in equation (2), the only case that $C_{i,t}^{A} = 1$ and $C_{j,f(t)}^{B} = 1$ is when $a_t = x_i$ and $b_{f(t)} = x_j$ and so $d\left(a_t, b_{f(t)}\right) = d\left(x_i, x_j\right)$.

Properties

Property a) The distance measure *D(A,B)* between sets *A* and *B* satisfy the metric properties.

Property b) The distance value of distances between sets and histograms of these sets is the same, *D(A,B) = D(H(A),H(B))*.

Property c) The distance measure *D(H(A),H(B))* between histograms *H(A)* and *H(B)* satisfy the metric properties.

Proofs

Property (a): The proof of this property was depicted in [5]. Although in that paper, the histograms were defined one-dimensional, the proof was based on the distance between elements *d(a,b)* independently on the dimension of the elements *a* and *b*.

Property (b): If we apply equation (10) to substitute the distance between elements $d\left(a_t, b_{f_{opt}(t)}\right)$ in the definition of the distance between sets (6), we obtain the formula $D(A,B) = \sum_{t=1}^{n}\sum_{i,j=1}^{z} C_{i,t}^{A} C_{j,f_{opt}(t)}^{B} d\left(x_i, x_j\right)$. Then, rearranging the elements, we get $\sum_{i,j=1}^{z} d\left(x_i, x_j\right) \sum_{t=1}^{n} C_{i,t}^{A} C_{j,f_{opt}(t)}^{B}$.

Finally, if we substitute the equation of the flow (9) we obtain the final expression, $\sum_{i,j=1}^{z} d(x_i, x_j) g_{f_{opt}}(i,j) = D(H(A), H(B))$.

Property (c): The proof is simple since we have proved that the distance value is the same (property b) and that the distance measure between sets satisfy the metric property (property a).

4 Algorithm

In this section, we depict an efficient algorithm used to compute the distance between histograms based on a solution to the well-known transportation problem [3]. Suppose that several suppliers, each with a given amount of goods, are required to supply several consumers, each with a given limited capacity. For each pair of suppliers and consumers, the cost of transporting a single unit of goods is given. The transportation problem is then to find a least-expensive flow of goods from the suppliers to the consumers that satisfies the consumer's demand. Our distance between histograms can be naturally cast as a transportation problem by defining one histogram as the supplier and the other one as the consumer. The cost of transporting a single unit of goods is set to the distance between the *bin* of one histogram and the *bin* of the other one, $d(x_i,x_j)$. Intuitively, the solution of the transportation problem, $g_f(i,j)$, is then the minimum amount of "work" required to transform one histogram to the other one subjected to the constraints defined by the properties of the flow $g_f(i,j)$ (section 4.2.2).

The computational cost of the transportation problem is exponential, respect the number of suppliers and consumers, that is, the number of bins of the histograms, z. Fortunately, efficient algorithms are available. One of the most common solutions is the simplex algorithm (), which is an iterative method that the cost of one simplex iteration is $O(z^2)$. The main drawback is that the number of iterations is not bounded and that this method needs a good initial solution. The Russell method [4] is the most common method used to find the first solution with a computational cost of $O(2z\text{-}1)$.

In this paper, we present an efficient and not iterative algorithm (figure 1) with a computational cost of $O(2z\text{-}1)$.

Given a pair of bins from both histograms, i and j, our algorithm finds the amount of goods that can be transported, $g_f(i,j)$, and computes the cost of this transportation, $g_f(i,j)*d(x_i,x_j)$. The algorithm finishes when all the goods have been transported, that is, all the elements of the sets, n, have been considered. In each iteration, a pair of bins is selected by the function *next*, in a given order and considering that the bins are not empty. The order of the bins is set by the following energy function,

$$E(i,j) = Path\_Deviation_j(i) + Path\_Deviation_i(j) \tag{11}$$

The $Path\_Deviation_j(i)$ is the difference between the maximum cost from the bin i to any bin of the histogram and the real cost from this bin to the bin j,

$$Path\_Deviation_j(i) = \max\_dist(x_i) - d(x_i, x_j) \tag{12}$$

It represents the worst case that the good can be sent (supplier) or received (consumer) respect the best case.

```
Algorithm Histogram-Distance (H(A),H(B))
i,j = first()
while n > 0 // n: the number of elements of both sets
     $g_f(i,j)$ = min ($H_i(A)$ , $H_j(B)$)
     $H_i(A)$ = $H_i(A)$ - $g_f(i,j)$
     $H_j(B)$ = $H_j(B)$ - $g_f(i,j)$
     n = n - $g_f(i,j)$
     D = D + $g_f(i,j)$ * $d(x_i,x_j)$
     i,j = next (i , j , H(A), H(B))
Return D  //distance between histograms
```

Fig. 1. Algorithm that computes the distance between n-dimensional histograms

Theorem. The worst computational cost of the algorithm is *O(2z-1).*

Proof. The pair of bins i,j generated by the function *next* forms a *z X z* matrix. In each iteration, one column or file (or both) of the matrix (depending if $H_i(A) = 0$ or $H_j(B) = 0$ is erased from the matrix (can not be used any more). Then, the worst case is the one that alternatively, one column is erased and after that one file is erased. Thus, the number of iterations is the number of columns plus the number of files less one.

5 Experimental Validation

We have used the coil image database [7] to validate our new algorithm and to show the usefulness of the histograms as the only information of the images. Only 20 objects were selected (figure 2). The test set was composed by 100 images (5 images

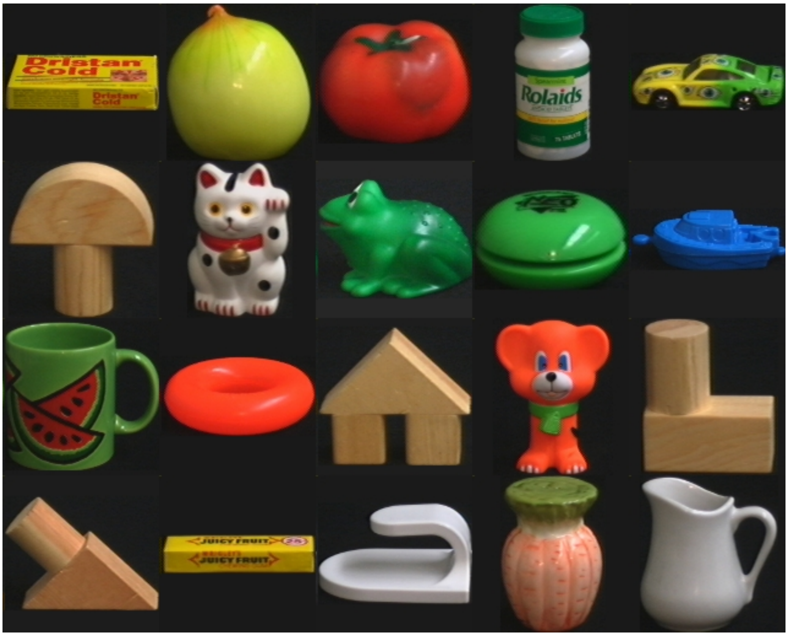

Fig. 2. Images taken at angle 5 of the 20 objects

of these 20 objects taken at the angles 5, 15, 25, 35 and 45). And the reference set was composed by other 100 images (5 images of the same objects taken at angles 0, 10, 20, 30, 40 and 50).

Table 1 (left) shows the number of correctly classified images (1-nearest neighbour) and (right) the average number of iterations of the inner loop of the algorithm in figure 1. The run time is proportional to the number of iterations. The first column is the number of bins (and bits) per each dimension. The number of colours is $bins^{nD}$. In the other columns, we show the results for 3 different 3D-histograms, 2 different 2D-histograms and 2 more 1D-histograms. The number of iterations underlined and in bold (right table) are the ones that all the images have been properly classified (99 or 100% in left table). If the recognition ratio is expected to be 99 or 100%, the best combination is HSV(2bits), CIELAB(3bits), HL(3bits) and HS(3bits).

Table 1. (left) Number of objects properly classified and (right) average number of iterations

Dimension	3D			2D		1D	
Bins(bits)	HSV	RGB	CIELAB	HS	HL	HUE	GREY
4 (2)	99	98	95	98	97	77	64
8 (3)	100	97	99	99	100	94	94
16 (4)	100	100	100	99	100	95	96
64 (6)	--	--	--	99	100	97	100
256 (8)	--	--	--	--	--	97	100

Dimension	3D			2D		1D	
Bins(bits)	HSV	RGB	CIELAB	HS	HL	HUE	GREY
4 (2)	53	32	19	20	20	6	6
8 (3)	**250**	120	55	70	**70**	14	13
16 (4)	**896**	**425**	**180**	219	**192**	29	26
64 (6)	--	--	--	1431	**1100**	95	**100**
256 (8)	--	--	--	--	--	229	**383**

Table 2 shows the worst number of iterations obtained from the theoretical cost. We realise that there is a huge difference between the real number of iterations (table 1 right) and the worst cases (table 2).

Table 2. Worst number of iterations obtained from the theoretical cost

	3D			2D		1D	
Bins (bits) X dimension	HSV	RGB	CIELAB	HS	HL	HUE	GREY
4 (2)	$2*4^3-1 = 127$			$2*4^2-1 = 31$		$2*4^1-1 = 7$	
8 (3)	$2*8^3-1 = 1{,}023$			$2*8^2-1 = 127$		$2*8^1-1 = 15$	
16 (4)	$2*16^3-1 = 8{,}191$			$2*16^2-1 = 511$		$2*16^1-1 = 31$	
64 (6)	$2*64^3-1 = 524{,}287$			$2*64^2-1 = 8{,}191$		$2*64^1-1 = 127$	
256 (8)	$2*256^3-1 = 33{,}554{,}431$			$2*256^2-1 = 131{,}071$		$2*256^1-1 = 511$	

6 Conclusions and Future Work

We have presented a new distance between multi-dimensional histograms and an efficient algorithm to compute this distance. Our method is useful for comparing black&white or colour images and using H,S,I or R,G,B colour descriptors. The theoretical computational cost is $O(2z)$, being z the number of levels of the pixels. The experimental validation demonstrates that it is worth increasing the number of dimensions and reducing the number of bins per each dimension, i.e. HSV (2bits).

Moreover, the real number of iterations (or run time) is really lower than the theoretical one.

References

1. S.-H. Cha, S. N. Srihari, “On measuring the distance between histograms” *Pattern Recognition* 35, pp: 1355–1370, 2002.
2. Y. Rubner, C. Tomasi, and L. J. Guibas, “A Metric for Distributions with Applications to Image Databases” *International Journal of Computer Vision* 40 (2), pp: 99-121, 2000.
3. *Numerical Recipes in C: The Art of Scientific Computing*, ISBN 0-521-43108-5.
4. E. J. Russell. “Extension of Dantzig's algorithm to finding an initial near-optimal basis for the transportation problem”, *Operations Research*, 17, pp: 187-191, 1969.
5. F. Serratosa & A. Sanfeliu, “Signatures versus Histograms: Definitions, Distances and Algorithms”, *Pattern Recognition* (39), Issue 5, pp. 921-934, 2006.
6. F.-D. Jou, K.-Ch. Fan, Y.-L. Chang, “Efficient matching of large-size histograms”, *Pattern Recognition Letters* 25, pp: 277–286, 2004.
7. http://www1.cs.columbia.edu/CAVE/research/softlib/coil-100.html