Pseudo-Euclidean Representation for Effective Multi-Class Texture Classification

A. K. C. Wong
H. C. Shen
P. W. Wong

Systems Design Engineering
University of Waterloo

Abstract

This paper describes how the theory of pseudo-Euclidean embedding can be applied for effective multi-class texture classification. The textures are classified according to the nearest neighbour rule based on our recently developed texture metric. Our objective is to significantly reduce the amount of computation from an exhaustive search scheme. For this purpose, a distance preserving vector space representation of the texture data base is constructed. This representation facilitates the selection and organization of a small subset of class prototypes so that the search for the nearest class can be carried out economically. This methodology is demonstrated by experiments on 720 texture samples belonging to eight classes. On average, a reduction of close to 70% is achieved.

Keywords: *pseudo-Euclidean spaces, distance preserving embedding, multi-class texture classification*

1 Introduction

The effectiveness of a supervised classification scheme depends on what distance function is used to discriminate the samples as well as how the search for the correct classes are conducted. For multi-class texture classification, we have recently developed a texture metric which has been demonstrated to be highly effective in discriminating perceptually similar textures of real objects[1,2]. In our original approach, classification is performed by simply computing the distances of the texture to be classified to *all* of the samples in the texture database. This exhaustive method is clearly un-economical when the database is large. In this article we will propose an improvement of the exhaustive method by addressing the issue of effective search. Our methodology employs a distance preserving vector space representation of the texture data. This representation facilitates the selection of a small subset of class prototypes which furnish all essential spatial information for nearest neighbour classification. Accurate and economical classifications can then be achieved by comparing against these prototypes alone.

In order to identify such prototypes, one must first have an *overall* picture of the database. If we could visualize our data geometrically, we would be able to describe them in terms of the relative locations of the classes, as well as their shapes and boundaries. By treating the texture elements as a finite metric set [4], we can *embed* them (i.e. represent them as points) in a pseudo-Euclidean vector space. The geometric configuration arising from the representation space would then provide the basis for the selection of prototypes as well as the development of a hierarchical classification scheme.

Though our present application is specifically on texture classification, it is in fact an illustration of a new approach to a very general class of problems. Provided we have a good metric, our methodology can be generally applied to any type of data which cannot originally be represented as pattern vectors in a feature vector space.

2 Pseudo-Euclidean Representation of Texture Data

For completeness, we will briefly describe the theory leading to our methodology. The reader is referred to the original references [3-6] for complete details.

2.1 Basic Concepts and Definitions

Let our texture metric be denoted by f. It has been shown in [2] that f satisfies the following properties:

reflexivity for all texture samples t_i, t_j, $f(t_i, t_j) \geq 0$ with equality iff $t_i = t_j$,

symmetry for all t_i, t_j, $f(t_i, t_j) = f(t_j, t_i)$,, and

triangle inequality for all t_i, t_j, t_k ,

$$f(t_i, t_k) \leq f(t_i, t_j) + f(t_j, t_k).$$

Let $S = \{t_0, \ldots, t_N\}$ be a set of $N+1$ texture samples. S, together with f, is referred to as a *finite metric set.*

Our objective is to represent $\mathcal{S}$ as a set of vectors in some vector space V so that *the distances between elements are preserved under the representation.* More precisely, we wish to find a distance preserving mapping

$$\mathcal{A} : \mathcal{S} \rightarrow V$$

such that

$$||\mathcal{A}(t_i) - \mathcal{A}(t_j)||_V = f(t_i, t_j) \; \forall t_i, t_j \in \mathcal{S}.$$

In the above mapping, V is *at most* an N dimensional vector space (since $\mathcal{S}$ has $N+1$ elements) and $||\cdot||_V$ is the vector distance function defined on V.

It is well known that in order to measure distances in a vector space, it must first be supplied with an inner product. In general, the familiar class of Euclidean inner product spaces cannot represent an arbitrary finite metric set. Instead, a larger class known as *pseudo-Euclidean spaces* is required. A pseudo-Euclidean space is a vector space V together with a real valued inner product $(.|.)$ such that for all $\vec{x}$, $\vec{y}$, $\vec{z}$ in V and all c in R,

1. $(\vec{x} + \vec{y}|\vec{z}) = (\vec{x}|\vec{y}) + (\vec{x}|\vec{z})$,
2. $(c\vec{x}|\vec{y}) = c(\vec{x}|\vec{y})$,
3. $(\vec{x}|\vec{y}) = (\vec{y}|\vec{x})$.

In standard coordinates, the inner product of $\vec{x}$ and $\vec{y}$ in an $N(= p+q)$-dimensional pseudo-Euclidean space can be expressed as

$$(\vec{x}|\vec{y}) = x_1 y_1 + \ldots + x_p y_p - x_{p+1} y_{p+1} - \ldots - x_{p+q} y_{p+q},$$

or equivalently,

$$(\vec{x}|\vec{y}) = \begin{bmatrix} x_1 \\ \vdots \\ x_{p+q} \end{bmatrix}^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_{p+q} \end{bmatrix},$$

where I_p and I_q are identity matrices of ranks p and q respectively. The square of the vector distance between $\vec{x}$ and $\vec{y}$ is defined as

$$||\vec{x} - \vec{y}||_V^2 := (\vec{x} - \vec{y}|\vec{x} - \vec{y}).$$

We will denote such an inner product space as $R^{(p,q)}$. (p,q) is known as its *vector signature.* The familiar N dimensional Euclidean space corresponds to the special case $p = N$ and $q = 0$.

The *existence* of a distance preserving embedding of a finite metric data set into a pseudo-Euclidean space follows from a theorem due to I. J. Schoenberg [3] and a full proof by L. Goldfarb [4]. Presently we will provide an equivalent proof which explicitly describe the construction process.

2.2 A Simple Constructive Proof

Our objective is to determine the signature (p,q) of the representative pseudo-Euclidean space and the coordinates of $\mathcal{A}(t_i)$, $0 \leq i \leq N$, in $R^{(p,q)}$. Let $\vec{x}_i \equiv (x_1^i, \ldots, x_N^i)$ be the *yet to be determined* coordinates of $\mathcal{A}(t_i)$. Without loss of generality, $\mathcal{A}(t_0)$ can always be chosen to be the origin, i.e. $\vec{x}_0 = (0, \ldots, 0)$. The distance preserving condition implies that for $0 \leq i \leq j \leq N$,

$$\begin{bmatrix} x_1^i - x_1^j \\ \vdots \\ x_N^i - x_N^j \end{bmatrix}^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \begin{bmatrix} x_1^i - x_1^j \\ \vdots \\ x_N^i - x_N^j \end{bmatrix} = f^2(t_i, t_j). \quad (1)$$

Equation (1) can be expanded and simplified so that for $1 \leq i, j \leq N$,

$$\begin{bmatrix} x_1^i \\ \vdots \\ x_N^i \end{bmatrix}^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \begin{bmatrix} x_1^j \\ \vdots \\ x_N^j \end{bmatrix} = M_{ij}, \quad (2)$$

where

$$(M)_{ji} := \frac{1}{2}\left\{ f^2(t_i, t_0) + f^2(t_j, t_0) - f^2(t_i, t_j) \right\}.$$

Note that M is symmetric because f is symmetric. In more compact form, equation (2) can be expressed as

$$X^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} X = M, \quad (3)$$

where X, with $(X)_{ji} = x_j^i$, is the matrix whose columns are the representation vectors for t_i.

Thus our problem amounts to factoring M into the form of the left hand side of equation (3). Since M is real and symmetric, a well known theorem in linear algebra assures that it can be factored (for example via the well known QR algorithm) as

$$M = E \; \mathrm{diag}\{\lambda_1 \ldots, \lambda_N\} \, E^t$$

with

$$E^t = E^{-1}.$$

In the above expression, λ_i stands for the ith eigenvalue of M and the ith column of E is the ith *unit* eigenvector, i.e.

$$e_{i1}^2 + \cdots + e_{iN}^2 = 1. \quad (4)$$

In general these eigenvalues can be either positive (including zero) or negative. p and q correspond to the numbers of positive and negative eigenvalues respectively. Without loss of generality, we can rearrange the coordinate system so that the first p eigenvalues are positive. The matrix $\mathrm{diag}\{\lambda_1 \ldots, \lambda_N\}$ can thus be further factored as

$$\mathrm{diag}\{\lambda_1 \ldots, \lambda_N\} = \Lambda \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \Lambda,$$

where

$$\Lambda := \mathrm{diag}\left\{ |\lambda_1|^{\frac{1}{2}}, \ldots, |\lambda_N|^{\frac{1}{2}} \right\}.$$

Combining the results of both factorizations,

$$X = \Lambda E^t. \quad (5)$$

We have thus constructed a representation for $\mathcal{S}$ in $R^{(p,q)}$ via the distance preserving embedding $\mathcal{A}$:

$$\mathcal{A}(t_0) = \text{the origin},$$

$$\mathcal{A}(t_i) = (|\lambda_1|^{\frac{1}{2}} e_{i1}, \ldots, |\lambda_N|^{\frac{1}{2}} e_{iN})^t, \; 1 \leq i \leq N. \quad (6)$$

Equation (4) implies that $|e_{ij}| \leq 1$ and it follows from equation (6) that the magnitude of *every* jth coordinate of $\vec{x}_i$ is at most $|\lambda_j|^{\frac{1}{2}}$. Hence dimensionality reduction can be achieved by dropping the dimensions associated with small eigenvalues. In particular, dimensions corresponding to zero eigenvalues can be discarded without any loss of accuracy.

With these ideas in mind, we now proceed to describe how the texture classifier is actually constructed.

3 Construction Procedure

Let our data set S be made up of classes $\mathbf{C}_0, \ldots, \mathbf{C}_K$. To formalize our discussion, we first introduce a few definitions.

Definition 1 *The* center, c_i, *of class* $\mathbf{C}_i$ *is the element of the class whose sum squared distances to all elements of* $\mathbf{C}_i$, *i.e.* $\sum_{\forall t \in \mathbf{C}_i} f^2(t, c_i)$, *is minimum.*

Definition 2 *The* radius *of* C_i *is* $\max_{t \in \mathbf{C}_i} f(t, c_i)$.

Definition 3 *The* intra-class distance *is* $\frac{1}{|\mathbf{C}_i|} \sum_{t \in \mathbf{C}_i} f(t, c_i)$.

Definition 4 *The* inter-class distance *between* $\mathbf{C}_i$ *and* $\mathbf{C}_j$ *is defined to be* $f(c_i, c_j)$.

3.1 Ranking Information Contained in the Data

In principle, no matter how large N (the size of S) is, an *exact* representation can always be constructed by the way described in the constructive proof. However, the dimensionality of the representation space would be very high because the distances between all pairs of elements are treated as equally important. The key to an *approximate* representation with low dimensionality lies in keeping only those dimensions arising from the subsets of distances which contribute significantly to the separation among the classes.

In order for an accurate and low dimensionality representation to be possible, the following assumption is necessary.

Assumption 1 *The metric* f *defined on* S *yields intra-class distances which are significantly smaller than the inter-class distances.*

Suppose an exact representation of S has indeed been constructed. Assumption 1 implies that the classes would form distinct clusters in the so constructed high dimensional space. After the insignificant dimensions are discarded, the large separations between clusters would be retained while minor variations within clusters would be smoothed out.

Next consider a non-center element t'; i.e. $t' \in S \setminus C$, where C is the set of class centers. Let c' be the center of the class to which t' belongs. Since by assumption the distance from t' to c' is small, it may be significantly altered by dimensionality reduction while the distances from t' to all other class centers should be relatively unaffected. Finally, let t'' be another non-center element. The exact distance from t'' to t' is restricted by the triangle inequality to within a limited range:

$$f(t'', c') - f(c', t') \leq f(t'', t') \leq f(t'', c') + f(c', t').$$

Note that the amount of uncertainty is *twice* the value of $f(c', t')$. Thus we expect in general that dimensionality reduction would cause the greatest changes in the distances between non-center elements, less so in the distances between centers and non-center elements and the least changes in the inter-class distances.

The above observations suggest that our data set, which can be conceived of as simply a collection of distances, be divided into three categories in decreasing order of importance:

1. $\mathcal{G}_1 := \{f(c, c') | \forall c, c' \in C\}$,
2. $\mathcal{G}_2 := \{f(t, c) | \forall t \in S \setminus C$ and $\forall c \in C\}$ and
3. $\mathcal{G}_3 := \{f(t, t') | \forall t, t' \in S \setminus C\}$.

$\mathcal{G}_1$ contains the most important information because in knowing the distances between the centers, the locations of the classes relative to one another can be estimated. $\mathcal{G}_2$ is of secondary importance in that it provides the information to position each non-center element relative to the centers. Intuitively, this enables us to determine the general shapes of the classes. Finally, since any distance $f(t'', t')$ in $\mathcal{G}_3$ lies in between $f(t'', c') \pm f(t', c')$, which are both distances in $\mathcal{G}_2$, $\mathcal{G}_3$ carries little additional information. More specifically, $\mathcal{G}_3$ contains the information about *exact* shapes of the classes which can be reconstructed at the expense of high dimensionality.

The above ranking suggests a natural way to 'build up' a representation starting with the most important components of a data set. Figure 1 gives an overview of the iterative embedding process.

3.2 Iterative Formation of Multi-Class Representation

As Figure 1 indicates, our basic approach is to first determine the positions of the class centers. The locations for the remaining elements are then determined one at a time.

3.2.1 Determine the Locations of the Centers

We begin by embedding, using the method described in Section 2.2, the $K+1$ class centers $c_0, \ldots, c_K$ into the vector space $R^{(p,q)}$ based on the information contained in $\mathcal{G}_1$. Let

$$\lambda_1 \geq \ldots \geq \lambda_p > 0 > \lambda_{p+1} \geq \ldots \geq \lambda_{p+q} \quad (p + q \leq K)$$

be the eigenvalues obtained in this construction and let $\vec{x}^i$, $0 \leq i \leq K$, be the representation vectors for the class centers. We have assumed, without any loss of generality, that all the eigenvalues are non-zero. This preliminary representation will be referred to as the *skeleton* upon which

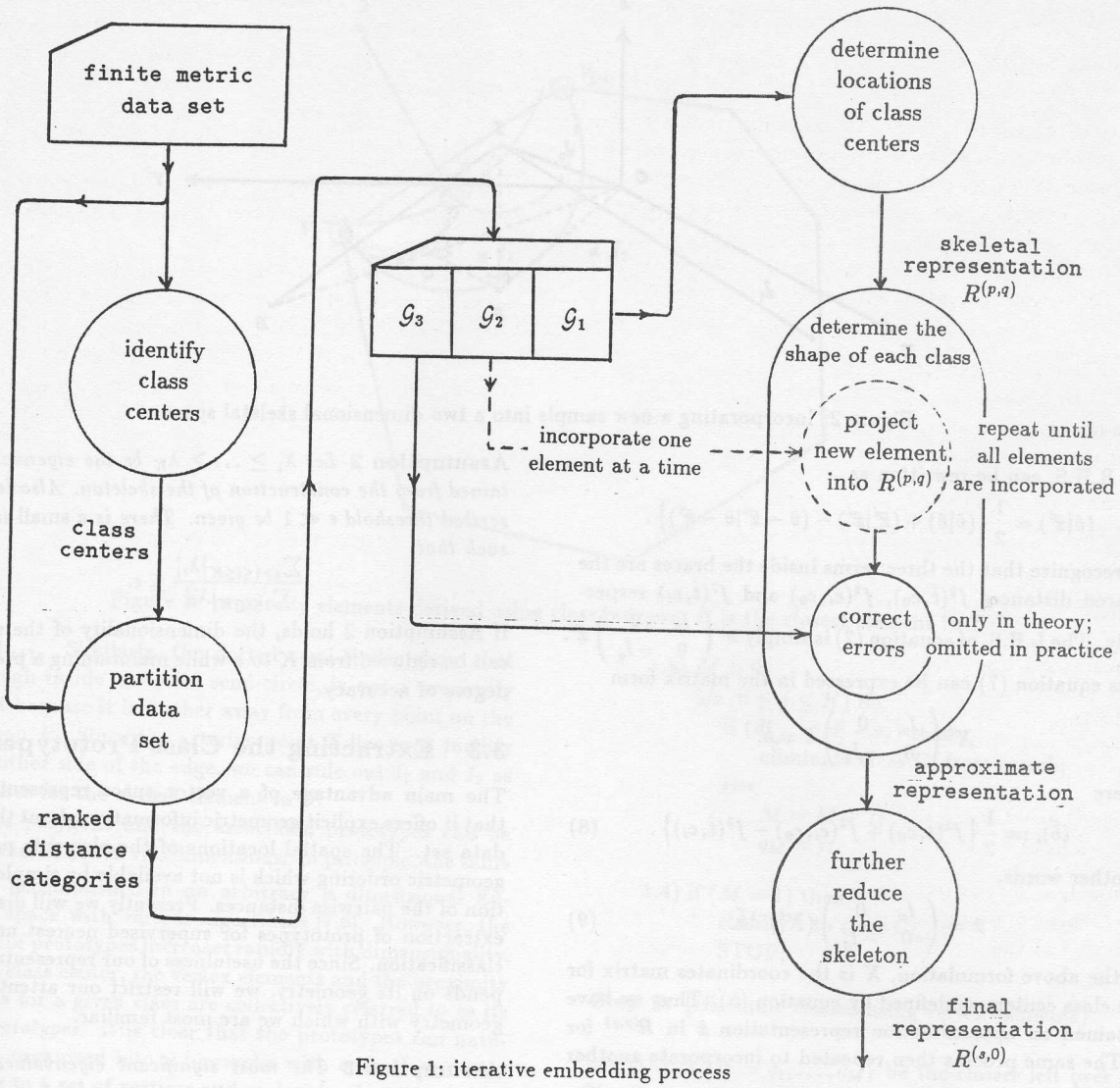

Figure 1: iterative embedding process

additional details concerning the various classes will be attached.

3.2.2 Determine the Shape of Each Class

Next, we 'fill in' the shape of each class by successively embedding the remaining elements around their respective class centers. Let $\hat{t}$ be the current element to be embedded. From $\mathcal{G}_2$, we know the distances from $\hat{t}$ to the class centers. Geometrically, these $K+1$ distances determine the position of $\hat{t}$ in a $K+1$ dimensional space which contains the skeleton as a K dimensional subspace. The component in the $(K+1)$th dimension is negligible because $\hat{t}$ is much closer to its class center than to the other centers. We can therefore obtain an approximate representation of $\hat{t}$ by dropping the $(K+1)^{\text{th}}$ dimension. This process is known as the *orthogonal projection* of $\hat{t}$ into the skeletal space.

Figure 2 illustrates this idea with a two dimensional skeleton. Points O, A and B are respectively the locations of class centers c_0, c_1 and c_2 on the $X-Y$ plane. The distances $\hat{d}_0$, $\hat{d}_1$ and $\hat{d}_2$ from $\hat{t}$ to the centers fix the position of $\hat{t}$ to be at T. P is the projection of T onto the plane. $|TP|$, the length of the component in the Z dimension, can be estimated by $|TR|$, the height of $\triangle OBT$. (Alternatively, we can also use $\triangle OAT$.) Clearly $|TR|$ is insignificant if $|OB| \gg |OT|$. In this case, P would be a good approximate representation for $\hat{t}$.

In general, let $\hat{v}$ (illustrated by $\vec{OT}$ in Figure 2) be the exact vector representation of $\hat{t}$ in some higher dimensional space, and $\hat{x}$ ($\vec{OP}$) be the projection of $\hat{v}$ onto the skeletal space $R^{(p,q)}$. It can be easily verified that $\hat{x}$ and $\hat{v}$ have the same orthogonal projection ($\vec{OQ}$) on $\vec{x}^i$ ($\vec{OB}$), for $1 \leq i \leq p+q$. Since the orthogonal projection between two vectors is given by their inner product,

$$(\hat{x}|\vec{x}^i) = (\hat{v}|\vec{x}^i). \tag{7}$$

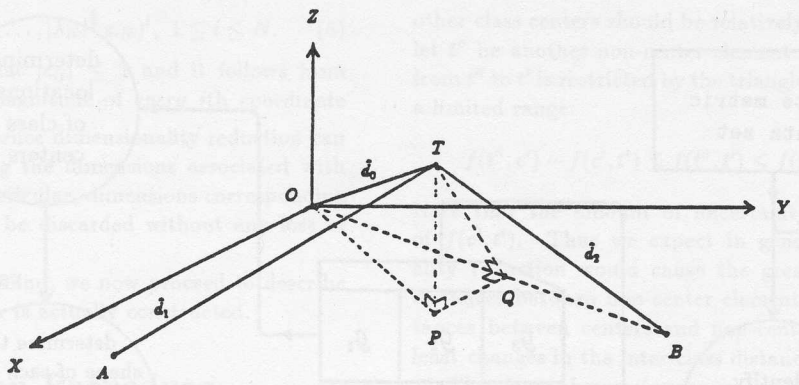

Figure 2: incorporating a new sample into a two dimensional skeletal space

The R.H.S. can be rewritten as

$$(\hat{v}|\vec{x}^i) = \frac{1}{2}\left\{(\hat{v}|\hat{v}) + (\vec{x}^i|\vec{x}^i) - (\hat{v}-\vec{x}^i|\hat{v}-\vec{x}^i)\right\}.$$

We recognize that the three terms inside the braces are the squared distances $f^2(\hat{t},c_0)$, $f^2(c_i,c_0)$ and $f^2(\hat{t},c_i)$ respectively. The L.H.S. of equation (7) is simply $\hat{x}^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \vec{x}^i$. Thus equation (7) can be expressed in the matrix form

$$X^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} \hat{x} = \vec{b},$$

where

$$(\vec{b})_i := \frac{1}{2}\left\{f^2(\hat{t},c_0) + f^2(c_i,c_0) - f^2(\hat{t},c_i)\right\}. \qquad (8)$$

In other words,

$$\hat{x} = \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} (X^t)^{-1}\vec{b}. \qquad (9)$$

In the above formulation, X is the coordinates matrix for the class centers as defined by equation (5). Thus we have obtained an approximation representation $\hat{x}$ in $R^{(p,q)}$ for $\hat{t}$. The same process then repeated to incorporate another element.

Physically the projection of elements into the skeletal space can be interpreted as a 'local flattening' of the actual representation space around the class centers. This approximation causes an *squared representation error* $\epsilon(t_i,t_j)$ in representing the distance between elements t_i and t_j:

$$f^2(t_i,t_j) - (\mathcal{A}(t_i) - \mathcal{A}(t_j))^t \begin{pmatrix} I_p & 0 \\ 0 & -I_q \end{pmatrix} (\mathcal{A}(t_i) - \mathcal{A}(t_j)).$$

In the above expression, $\mathcal{A}(t)$ is the vector representation of t and $f(t_i,t_j)$ is of course contained in $\mathcal{G}_3$. Thus $\mathcal{G}_3$ is useful only in determining the errors in the representation and does not enter explicitly in the construction process.

3.2.3 Further Reduction of the Skeleton

At this stage, we have already significantly reduced the dimensionality because $K \ll N$. However, it is very advantageous to make the dimensionality of the final representation as small as possible. Thus we would seek to further reduce the skeletal space. For this we need another assumption.

Assumption 2 *Let $\lambda_1 \geq \ldots \geq \lambda_K$ be the eigenvalues obtained from the construction of the skeleton. Also let a prescribed threshold $\epsilon \ll 1$ be given. There is a* small *integer s such that*

$$\frac{\sum_{s+1\leq i\leq K}|\lambda_i|}{\sum_{1\leq i\leq K}|\lambda_i|} \leq \epsilon.$$

If Assumption 2 holds, the dimensionality of the skeleton can be reduced from K to s while maintaining a prescribed degree of accuracy.

3.3 Extracting the Class Prototypes

The main advantage of a vector space representation is that it offers explicit geometric information about the whole data set. The spatial locations of the elements provide a geometric ordering which is not available by simple inspection of the pairwise distances. Presently we will discuss the extraction of prototypes for supervised nearest neighbour classification. Since the usefulness of our representation depends on its geometry, we will restrict our attention to a geometry with which we are most familiar.

Assumption 3 *The most significant eigenvalues associated with the skeleton are all positive.*

This will ensure that the final representation space is Euclidean.

With an Euclidean representation, we can define the *class polyhedron* Δ_i to be the smallest convex polyhedron which contains all the elements in class $\mathbf{C}_i$. The set $\mathcal{V}_i$ of *vertex elements* consists of all the elements located at the vertices of Δ_i. For each point on each face of the polyhedron, we find the class element closest to it. The collection of all such elements is identified as the set of *proximity elements* to the surface of the Δ_i. The closest element in $\mathbf{C}_i$ to any given point T outside Δ_i must clearly be one of the vertex or proximity elements.

Figure 3 illustrates this idea in two dimensions. The solid semi-circle is centered at the midpoint of the edge connecting vertices V_j and V_{j+1}. The set of all potential proximity elements to that edge, denoted as $\mathcal{P}_{j,j+1}$, must lie inside this semi-circle because any other element, e.g. I_2, is clearly further away from every point on that edge than

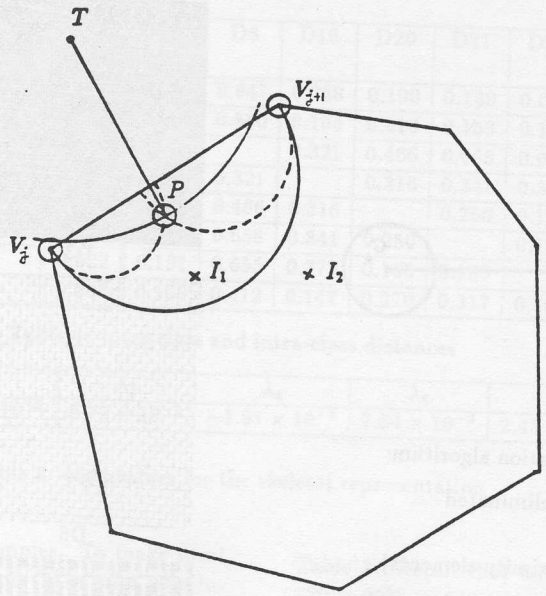

Figure 3: proximity elements derived using class polygon; P is the closest element to T

V_j or V_{j+1}. Similarly, the dotted semi-circles show that I_1, though inside the solid semi-circle, is *not* a proximity element because it is further away from every point on the edge than P. Since the exterior point T lies even further to the other side of the edge, we can rule out I_1 and I_2 as candidates for the closest element to T.

Class polygons and the associated prototypes can be easily determined in two dimensions. In principle, the same can be accomplished in an arbitrary p dimensional Euclidean space with various algorithms [7,8]. However, the number of prototypes increases rapidly with dimensionality.

The class center, the vertex elements and the proximity elements for a given class are collectively referred to as its *class prototypes*. It is clear that the prototypes can naturally be organized into a hierarchy with each class center pointing to a set of vertices and each pair of vertices in the same class pointing to a set of proximity elements. Using these prototypes, a 3-step search strategy can be developed.

4 Search-Effective Strategy

For simplicity, we will present our strategy for a two dimensional representation space. Let $\tilde{t}$ be the unknown texture to be classified.

begin{classification algorithm}

Step 1: {eliminate classes using hyperspheres}

1.1) for $(0 \leq i \leq K)$ do
compute from frequency diagrams of $\tilde{t}$
and c_i the distance $d_i^c := f(\tilde{t}, c_i)$;

1.2) find $j_{\min}$ such that $d^c_{j_{\min}} = \min_{0 \leq i \leq K} d_i^c$;

1.3) $M = 0$;
for $(0 \leq j \leq K)$ do
if $(d^c_{j_{\min}} < d_j^c - r_j)$ then
eliminate class C_j from search;
else
$M = M + 1$;
$q_M = j$;

1.4) if $(M = 1)$ then
classify $\tilde{t}$ to class $C_{j_{\min}}$;
STOP;

Step 2: {eliminate classes using vertex elements}

Let $\mathcal{Q} = \{\mathrm{C}_j | j = q_1, \ldots, q_M\}$ be the classes left over from Step 1.

2.1) for all $\mathrm{C}_j \in \mathcal{Q}$ and all $v_k^j \in \mathcal{V}_j$
compute $d^v_{j,k} := f(\tilde{t}, v_k^j)$;

2.2) identify $j_{\min}$ and $k_{\min}$ such that
$d^v_{j_{\min},k_{\min}} = \min_{j,k} d^v_{j,k}$;

2.3) $L = 0$;
for all $\mathrm{C}_j \in \mathcal{Q}$ and all $v_k^j \in \mathcal{V}_j$
if $(d^v_{j_{\min},k_{\min}} < d^v_{j,k} - f(v_k^j, v_{k+1}^j))$
eliminate the edge joining v_k^j and
v_{k+1}^j from search $(v^j_{|V_j|+1} \equiv v_1^j)$;
else
$L = L + 1$;
$R_L = j$;
$E_L = k$;

2.4) if $R_1 = \ldots = R_L = j_{\min}$ then
classify $\tilde{t}$ to class $\mathrm{C}_{j_{\min}}$;
STOP;

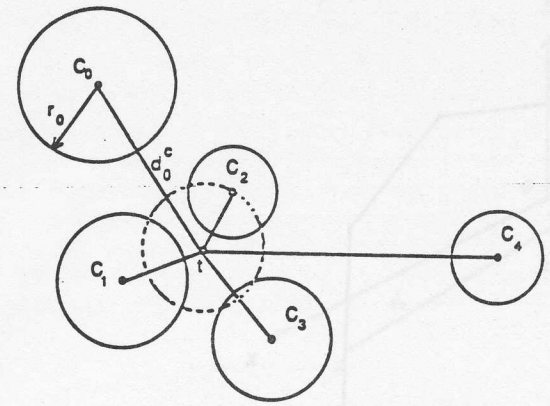

Figure 4: Step 1 in classification algorithm:

$j_{\min} = 2$; $\mathbf{C}_0$ and $\mathbf{C}_4$ are eliminated

Step 3: { Compare against proximity elements}

3.1) for all $l \in [1, L]$ and all $p_s \in P^{R_l}_{E_L, E_L+1}$ compute $d_{R_l,E_l,s} = f(\tilde{t}, p_s)$;

3.2) identify $\min_{R_l,E_l,s} d_{R_l,E_l,s}$ and record corresponding class number $l_{\min}$;

3.3) classify $\tilde{t}$ to class $\mathbf{C}_{l_{\min}}$;
STOP;

end{classification algorithm}

Twice in this algorithm, the triangle inequality is used to eliminate prototypes from further consideration. For example, Step 1.3 is illustrated by Figure 4 and is briefly explained as follows. The texture distance between $\tilde{t}$ and class $\mathbf{C}_i$ is

$$d(\tilde{t}, \mathbf{C}_i) := \min_{t \in \mathbf{C}_i} f(\tilde{t}, t) = f(\tilde{t}, t_{i_{\min}}),$$

where $t_{i_{\min}}$ is the element in $\mathbf{C}_i$ closest to $\tilde{t}$. According to the nearest neighbour rule, $\tilde{t}$ is classified to $\mathbf{C}_j$ iff

$$d(\tilde{t}, \mathbf{C}_j) = \min_i d(\tilde{t}, \mathbf{C}_i).$$

Since f satisfies the triangle inequality,

$$d^c_i \equiv f(\tilde{t}, c_i) \leq d(\tilde{t}, \mathbf{C}_i) + f(t_{i_{\min}}, c_i) \leq d(\tilde{t}, \mathbf{C}_i) + r_i, \ 0 \leq i \leq K.$$

In other words,

$$d^c_i - r_i \leq d(\tilde{t}, \mathbf{C}_i), \ 0 \leq i \leq K.$$

Hence if

$$d^c_{j_{\min}} \leq d^c_i - r_i$$

then

$$d^c_{j_{\min}} \leq d(\tilde{t}, \mathbf{C}_i)$$

and we can eliminate from our search all such classes $\mathbf{C}_i$. The rationale for Step 2.3 is similar and will not be described in detail here.

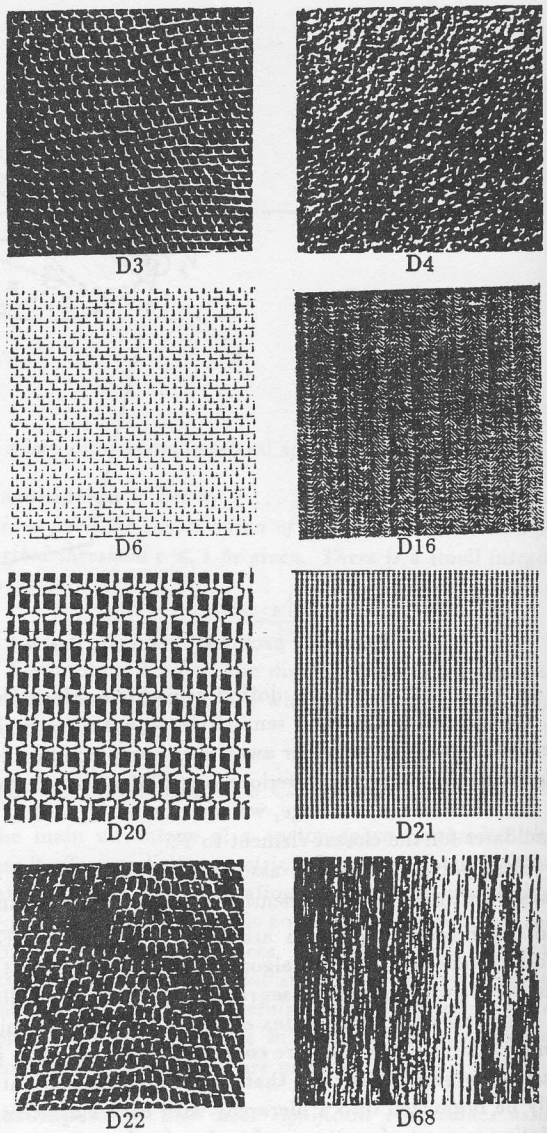

Figure 5: actual texture appearance

5 Experiments

Our methodology is tested eight classes of textures taken from Brodatz's album[9] (Figure 5). They are: reptile skin I (D3), pressed cork (D4), woven aluminum wire (D6), herringbone weave (D16), French canvas I (D20), French canvas II (D21), reptile skin II (D22), and wood grain (D68). Some of these classes are very similar. For example, woven aluminum wire (D6) is similar to French canvas II (D21) in terms of directionality, French canvas I (D20) is taken at 4 times the magnification of French canvas II (D21) and reptile skin I (D3) and reptile skin II (D22) only differ from one another in that they are photographed under different lighting conditions. The actual photographic images are digitized to produce the line and circular frequency diagrams.

intra-class distance	inter-class distance	D3	D4	D6	D16	D20	D21	D22	D68
0.054	D3		0.170	0.647	0.356	0.190	0.149	0.022	0.297
0.039	D4	0.170		0.500	0.194	0.218	0.153	0.191	0.180
0.077	D6	0.647	0.500		0.321	0.466	0.658	0.655	0.412
0.037	D16	0.356	0.194	0.321		0.316	0.341	0.372	0.147
0.054	D20	0.190	0.218	0.466	0.316		0.250	0.195	0.270
0.038	D21	0.149	0.153	0.658	0.341	0.250		0.169	0.317
0.044	D22	0.022	0.191	0.655	0.372	0.195	0.169		0.308
0.059	D68	0.297	0.180	0.412	0.147	0.270	0.317	0.308	

Table 1: inter-class and intra-class distances

λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
3.97×10^{-1}	5.42×10^{-2}	3.72×10^{-2}	-1.91×10^{-2}	7.34×10^{-3}	2.45×10^{-3}	8.06×10^{-4}

Table 2: eigenvalues for the skeletal representation

Altogether there are 720 texture samples. To make most efficient use of our data, the π-method[10] is used. Ten independent experiments are performed. In each experiment, 10% of the data is randomly withheld. The remaining texture elements are then used to construct the classifier. The proposed classification scheme is then used to classify the withdrawn samples. This procedure is repeated, i.e. with another subset randomly withheld and a new classifier constructed, until every texture sample has been withdrawn and tested.

5.1 Construction of the Classifiers

The classifiers are constructed according to the procedure outlined in Section 3. Table 1 displays the overall comparisons between the inter-class distances and the intra-class distances. With the exception of the two reptile skins (D3 and D22), all the intra-class distances are small comparing to the inter-class distances. Thus our data is in reasonable agreement with Assumption 1.

The eigenvalues arising from the pseudo-Euclidean embeddings do not vary significantly between different experiments. Table 2 displays the seven largest eigenvalues from one of the experiments. Only one of the eigenvalues is negative. Furthermore, the three most significant dimensions are Euclidean with

$$\frac{\sum_{i=4}^{7} |\lambda_i|}{\sum_{i=1}^{7} |\lambda_i|} \approx 5.8\%.$$

Thus Assumptions 2 and 3 are satisfied and we conclude that our data can be well approximated by a 3-dimensional Euclidean representation. In order to further simplify the computations, we decided to retain only 2 dimensions. The error incurred is still within acceptable limits:

$$\frac{\sum_{i=3}^{7} |\lambda_i|}{\sum_{i=1}^{7} |\lambda_i|} \approx 12.9\%.$$

5.2 Results and Evaluations

The results from the 10 experiments are summarized in Table 3. Recall that an exhaustive scheme would require $720 \times 90\% = 648$ comparisons for each sample classified. The high classification rate is not surprising since under an *one-to-all* comparison scheme, our texture metric provides a better than 98% classification rate for a similar set of experiments[2]. Further examinations of our current results show that 3 samples out of 90 in D3 (reptile skin I) are mis-classified to D22 (reptile skin II) and 3 samples in D21 (French canvas II) are mis-classified to D20 (French canvas I). Clearly, the 2-dimensional approximation has introduced errors in the discrimination among these two sets of very similar textures. Overall, our results do confirm that the Euclidean embedding is indeed a very accurate representation. The most important conclusion is that the proposed methodology does yield significant reduction in classification time (68.9%) as compared to an exhaustive method.

class	classification rate	average # of comparisons	% of exhaustive search
D3	96.7%	256.6	39.6
D4	100%	292.6	45.2
D6	100%	82.1	12.7
D16	100%	226.1	34.9
D20	100%	203.3	31.4
D21	96.7%	193.7	30.0
D22	100%	209.0	32.3
D68	100%	147.0	22.7
average percentage			31.1%

Table 3: π-test results

6 Conclusion

In summary, we have proposed a search-effective algorithm for multi-class texture classification. It is demonstrated that a vector space representation and a three steps search strategy lead to a significant reduction in the number of

comparisons. Provided Assumptions 1-3 hold, the pseudo-Euclidean embedding method provides an effective way to handle data of very general types. Our future research will investigate possible generalizations of our methodology under more relaxed conditions. We also hope to make further use of our texture metric by developing a hierarchical vector space representation so that an efficient data management scheme can be constructed for an even larger texture database.

References

[1] H.C. Shen and A.K.C. Wong, *Generalized Texture Representation and Metric*, Computer Graphics and Image Processing, Vol. 23, 1983, pp. 187-206.

[2] H. Shen, *A New Methodology for Multi-Class Texture Analysis and Classification*, Ph.D. dissertation, Dept. of Systems Design, University of Waterloo, Canada, 1982.

[3] I.J. Schoenberg, *Remarks to Maurice Frechet's Article*, Annals of Math. (2), Vol. 36, 1935, pp. 724-732.

[4] L. Goldfarb, *Riemannian Representation of Finite Symmetric Spaces and Its Applications To General Data Analysis*, Ph.D. dissertation, Dept. of Systems Design, University of Waterloo, Canada, 1979.

[5] H.C. Shen, L. Goldfarb and A.K.C. Wong, *Scatter Configuration, Typicality and Diversity of Ensembles of Textures*, Proc. on the International Conference on Cybernetics and Society, 1981, pp 176-181.

[6] L. Goldfarb, *A Unified Approach to Pattern Recognition*, Journal of Pattern Recognition, Vol. 17, No.5, 1984, pp. 575-582.

[7] D.R. Chand and S.S. Kapur, *An Algorithm for Convex Polytopes*, Journal of ACM, Vol. 17, No. 1, Jan. 1970, pp.78-86.

[8] J.L. Bentley, H.T. Kung, M. Schkolnick and C.D. Thompson, *On the Average Number of Maxima in a Set of Vectors and Applications*, Journal of ACM, Vol. 25, No. 4, Oct. 1978, pp. 536-543.

[9] P. Brodatz, Textures, Dover, New York, 1966.

[10] G.T. Toussaint, *Bibliography on Estimation of Misclassification*, IEEE Trans. on Information Theory, IT-20(4), 1974, pp. 472-479.