The Automatic Design of Feature Spaces for Local Image Descriptors using an Ensemble of Non-linear Feature Extractors

Gustavo Carneiro*
Instituto de Sistemas e Robótica
Instituto Superior Técnico, Lisbon, Portugal

Abstract

The design of feature spaces for local image descriptors is an important research subject in computer vision due to its applicability in several problems, such as visual classification and image matching. In order to be useful, these descriptors have to present a good trade off between discriminating power and robustness to typical image deformations. The feature spaces of the most useful local descriptors have been manually designed based on the goal above, but this design often limits the use of these descriptors for some specific matching and visual classification problems. Alternatively, there has been a growing interest in producing feature spaces by an automatic combination of manually designed feature spaces, or by an automatic selection of feature spaces and spatial pooling methods, or by the use of distance metric learning methods. While most of these approaches are usually applied to specific matching or classification problems, where test classes are the same as training classes, a few works aim at the general feature transform problem where the training classes are different from the test classes. The hope in the latter works is the automatic design of a universal feature space for local descriptor matching, which is the topic of our work. In this paper, we propose a new incremental method for learning automatically feature spaces for local descriptors. The method is based on an ensemble of non-linear feature extractors trained in relatively small and random classification problems with supervised distance metric learning techniques. Results on two widely used public databases show that our technique produces competitive results in the field.

1. Introduction

Local image descriptors have received an enormous attention ever since the seminal works of Schmid and Mohr [17] and Lowe [14]. Essentially, the design of local image descriptors is based on image features extracted from compact image regions (covering a small percentage of the image area) using certain types of spatial pooling methods (*e.g.*, weighted averaging, histogramming, *etc.*), which generates a feature transform. This feature transform can be mathematically [8, 13] or empirically [2, 14] shown to be robust to certain image deformations and to achieve a good trade-off between discriminating power and robustness. There are two distinct ways to understand the problem of designing feature transforms to be used by local image descriptors methods. The first approach is to come up with descriptors that show robustness to certain image deformations and present a reasonable discriminating power, and then find the matching problems that can be successfully solved with the use of such descriptors [2, 14, 17]. The second way is to start with a matching problem and search for feature transforms that can handle the deformations presented by the specific problem while producing descriptors that are discriminating enough [20].

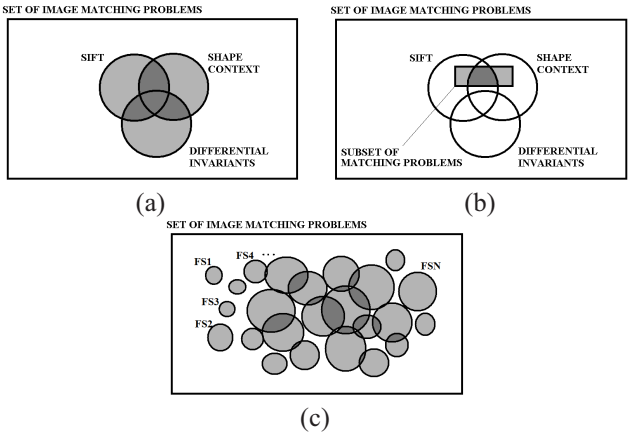


Figure 1. Set of image matching problems and the subset of problems solved by each type of feature transform (a). In Figure (b), we start with a specific matching problem (a subset of the original set) and search for the best descriptors (or a combination of descriptors) for the task. Figure (c) displays the combination of feature spaces proposed in this paper, where the union represents the whole set of matching problems that can be solved.

Figure 1-(a) shows a Venn diagram representing the whole set of image matching problems (rectangle) and the subset (circle) of problems solved by the feature transforms SIFT [14], shape context [2], and differential invariants [17]. This diagram means that each feature presents a good trade off between discriminating power and ro-

*This work was supported by the FCT (ISR/IST plurianual funding) through the PIDDAC Program funds and Project PRINTART (PTDC/EEA-CRO/098822/2008). This work was partially funded by EU Project IMASEG3D (PIIF-GA-2009-236173).

bustness for all problems located in its respective subset. There have been reports describing the types of matching problems that can be handled by certain image descriptors. For instance, Mikolajczyk [15] noticed that shape context presents high performance except for textured scenes (*e.g*., tree bark, brick wall) and when edges are not reliable. Also, Ke [12] presented PCA-SIFT as an alternative to SIFT for matching problems containing substantial image deformation (geometric transformation and brightness variations), which also indicates that SIFT-based descriptors are less suitable when large image deformations are present in the problem. Although a common goal of the authors of new types of local descriptors is to demonstrate their applicability in several matching problems, it is inevitable that each one of those descriptors can only be applied to a subset of the matching problems. Therefore it is unlikely that any of the current local descriptors will be useful for all types of matching problems. We call the feature transform that produces local descriptors which are useful to solve a large number of unanticipated matching problems, the universal feature transform.

In order to increase the set of matching problems solved by one type of descriptor, a quite simple solution is to form a new descriptor based on the combination of different descriptors (*e.g*. combine SIFT with shape context) [4, 16], where the hope is that the final subset of problems covered by the new descriptor will be represented by the union of their subsets, but unfortunately there is no guarantee that this is the case. The problem of combining local descriptor types has been elegantly posed as an optimization problem by Varma and Ray [20], where a combination of several descriptors is found in order to maximize the margin among descriptors produced by different local image classes for a specific matching problem (Fig. 1-(b)). Even though the latter solution is interesting, small changes to the subset (of the matching problem) require a new training process, which reduces the interest in the method for the general problem of universal feature transform unless it is possible to find descriptors covering the whole set of matching problems, but it remains to be seen whether this is a tractable optimization problem.

The main inspiration for this work is the paper by Hua *et al*. [11], where the authors propose a universal feature transform through an automatic feature selection process, which determines the type of feature to use and the spatial pooling method. Our main criticism is that this is still constrained to form one feature space, resulting in the same limitations mentioned before, but note that the set of problems covered by such feature space may be larger than any of the existing spaces. Also, the authors resort to the use of linear distance metric learning techniques, which we believe to be in the right direction except for the fact that they are linear, resulting in feature spaces with stable classifiers (the type of classifier used is specified later) with high bias and low variance, and consequently not useful for producing an ensemble of classifiers [3]. Implicitly, Hua *et al*. [11] acknowledge the limitation of the linear transform by first applying a predetermined non-linear transformation, which results in better feature spaces for classifying local descriptors. Another important reference is the work by Chopra *et al*. [6], which runs a learning procedure of a feature transform using only a subset of the test classes for a face verification problem. It remains to be investigated how this method would perform under the same experimental conditions suggested by Hua *et al*. [11], which is the main benchmark used in our paper.

In this work, we introduce a new method to solve the problem of designing a feature transform that can be used by a large number of unforeseen matching problems. We call it the Universal Feature Transform (UFT). The approach is defined as an incremental method for automatically learning feature spaces for local descriptors based on an ensemble of feature extractors. Each feature extractor is trained in relatively small and random classification problems using supervised non-linear distance metric learning techniques. By random, we mean that we train each feature space with a set of training classes selected randomly from a pool of training classes. The aggregation of the feature spaces is based on a simple distance sum scheme. Therefore, given a new matching problem, the distance between two descriptors is computed by the sum of the distances in all automatically learned feature spaces. Fig. 1-(c) shows the idea through the combination of several feature spaces (FS1 to FSN) trained with the method proposed in this paper, where the union of the FSi subsets represents the subset of matching problems solved by our approach. Experiments display simple examples that clarify the functionality of the method, and then we show results in the database of local descriptors provided by Winder and Brown [22], where we follow a training approach specific to our method, but use the same test conditions. Our results are comparable to the best results shown by the same group [11], but note that our method uses only the original gray values of the input image patch instead of pre-determined features and spatial pooling schemes to train the feature transformation, which means that the run-time complexity can be smaller with a clever implementation based on parallel algorithms. Moreover, the fact that we do not use pre-determined features has the potential to increase the subset of matching problems covered by our method. Using the feature space trained in the database above, we show that the matching results of our method in the problems developed by Mikolajczyk [15] are better than for current state-of-the-art local descriptors.

2. The Universal Feature Transform

The proposed universal feature transform (UFT) is an incremental learning method that can adapt itself to new matching problems, but that does not worsen its performance in problems that it already shows good performance, as new learning rounds are processed. We assume the existence of a large pool of training classes containing labeled local descriptors, where each class is produced with image regions taken from the same 3-D location in a scene (but from different viewpoints and viewing conditions) and aligned to a canonical image space (with similar orientation, scale and translation parameters). In order to assess quantitatively the accuracy of the method, we assume the

existence of a labeled test set built in the same way as the training set, but note that the intersection between the training and test sets is empty. The universal feature space presented here is an ensemble of feature spaces produced by non-linear supervised distance metric learning methods trained over random training sets of relatively small size.

2.1. Supervised Non-linear Distance Metric Learning

Our work is rooted in the supervised distance metric learning problem, which automatically designs feature spaces that bring closer together points belonging to the same class and that push farther apart points from different classes. Hence, this new distance metric has the potential to improve the accuracy of similarity-based classifiers [6, 9, 21], which is usually the type of classifier used in local descriptor matching methods. More specifically, our classification scheme is based on the threshold matching [15], where two descriptors (or two points) are matched if their distance in the transformed feature space is below a threshold. Following the notation used by Weinberger and Saul [21], let us first consider how to solve the linear distance metric learning [5, 10].

Assume that we have N image patches $\mathbf{x} \in \Re^n$ and respective labels $y \in \{1, ..., C\}$ forming the set $\mathcal{R} = \{(\mathbf{x}_i, y_i)\}_{i=1..N}$. A linear transform is represented by a matrix $\mathbf{T} \in \Re^{n \times m}$, where $m \leq n$ such that $\tilde{\mathbf{x}} = \mathbf{T}^\top \mathbf{x}$, $\tilde{\mathbf{x}} \in \Re^m$ and $\mathbf{T}^\top$ means the transpose of matrix $\mathbf{T}$. The distance between two points in the transformed space is then denoted as:

$$D_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) \quad = (\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j), \tag{1}$$

where $\mathbf{M} = \mathbf{T}\mathbf{T}^\top$. The convex optimization problem to learn the linear distance metric can be formulated as follows:

$$\begin{array}{rl} \text{minimize}_{\mathbf{M}} & \sum_{ij} \mathbf{Y}_{ij} \sqrt{D_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)} \\ \text{subject to} & \sum_{ij} (1 - \mathbf{Y}_{ij}) D_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) \geq 1 \\ & \mathbf{M} \succeq 0, \end{array} \tag{2}$$

where the element (i, j) of matrix $\mathbf{Y}$ is denoted as $\mathbf{Y}_{ij} = 1$ if $y_i = y_j$, and $\mathbf{Y}_{ij} = 0$ otherwise. In order to adapt this problem to a non-linear transformation, we first reformulate the problem (2) as follows [5, 18]:

$$\mathbf{T}^* = \arg \max_{\mathbf{T} \in \Re^{m \times n}} \left[\text{tr} \left(\left(\mathbf{T}^\top \mathbf{S}^{(w)} \mathbf{T} \right)^{-1} \left(\mathbf{T}^\top \mathbf{S}^{(b)} \mathbf{T} \right) \right) \right], \tag{3}$$

where

$$\begin{array}{l} \mathbf{S}^{(w)} = \frac{1}{2} \sum_{ij} \mathbf{W}^{(w)}_{ij} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top, \\ \mathbf{S}^{(b)} = \frac{1}{2} \sum_{ij} \mathbf{W}^{(b)}_{ij} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top, \end{array}$$

with $\mathbf{W}^{(w)} = \mathbf{Y}$ and $\mathbf{W}^{(b)} = 1 - \mathbf{Y}$ with $\mathbf{Y}$ defined in (2) (note that problems (2) and (3) are related, but not equivalent). Alternatively, this problem can be formulated as follows:

$$\begin{array}{rl} \text{minimize}_{\mathbf{T}} & -\frac{1}{2} \mathbf{T}^\top \mathbf{S}^{(b)} \mathbf{T} \\ \text{subject to} & \frac{1}{2} \mathbf{T}^\top \mathbf{S}^{(w)} \mathbf{T} = \mathbf{I}, \end{array} \tag{4}$$

where $\mathbf{I}$ denotes the identity matrix. Solving the dual of (4), we arrive at the following generalized Eigenvalue problem:

$$\mathbf{S}^{(b)} \mathbf{T} = \lambda \mathbf{S}^{(w)} \mathbf{T}, \tag{5}$$

where the eigenvectors associated with the m largest eigenvalues will form the linear transform $\mathbf{T}$.

The kernelization of the method, which turns it into a non-linear feature transform, is important since, intuitively, linear feature transforms are unlikely to handle the wide range of image deformations that typically appear in matching problems, and are also unlikely to increase the discriminating power of the original feature space. Also, even though we are dealing with feature transforms, it is important to recall Breiman's argument about ensemble classifiers. Breiman [3] argued that ensemble classifiers are effective only with the combination of unstable classifiers (low bias and high variance), which is likely to reduce the variance of the final classifier (and keep the low bias). Below, we show that similarity-based classifiers applied in non-linearly feature transformed spaces produce unstable results with low bias and high variance. Hence, using a similar reasoning introduced by Breiman, we propose an ensemble feature transform aggregating non-linear feature transforms in order to keep the low bias and reduce the variance as more feature transforms are added to the ensemble.

The kernelization of the approach [5, 18] is achieved by first observing that $\mathbf{S}^{(w)}$ and $\mathbf{S}^{(b)}$ in (3) can be written as follows:

$$\begin{array}{l} \mathbf{S}^{(.)} = \sum_i \left(\sum_j \mathbf{W}^{(.)}_{ij} \right) \mathbf{x}_i \mathbf{x}_i^\top - \sum_{ij} \mathbf{W}^{(.)}_{ij} \mathbf{x}_i \mathbf{x}_j^\top, \text{ or} \\ \mathbf{S}^{(.)} = \mathbf{X} \mathbf{L}^{(.)} \mathbf{X}^\top, \end{array} \tag{6}$$

where $\mathbf{L}^{(.)} = \mathbf{D}^{(.)} - \mathbf{W}^{(.)}$ with $\mathbf{D}^{(.)}_{ii} = \sum_j \mathbf{W}^{(.)}_{ij}$ being a diagonal matrix, and $\mathbf{X} \in \Re^{n \times N}$ is a matrix containing all the training points. Another observation is that $\mathbf{X}^\top \mathbf{T} = \mathbf{X}^\top \mathbf{X} \mathbf{U} = \mathbf{K} \mathbf{U}$, where $\mathbf{U} \in \Re^{N \times m}$ and with the element (i, j) of matrix $\mathbf{K} \in \Re^{N \times N}$ denoted as $\mathbf{K}_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$. Therefore, the generalized eigenvalue problem in (5) can be re-written as follows:

$$\mathbf{K} \mathbf{L}^{(b)} \mathbf{K} \mathbf{U} = \tilde{\Lambda} \mathbf{K} \mathbf{L}^{(w)} \mathbf{K} \mathbf{U} \tag{7}$$

by taking the equality $\mathbf{X}^\top \mathbf{T} = \mathbf{K} \mathbf{U}$ described above, and multiplying by $\mathbf{X}^\top$ on both sides. Therefore, $\{\mathbf{x}_i\}_{i=1..N}$ appear in terms of their inner product, and the non-linear transformation can be obtained using the *kernel trick* [19], with, for example, the following kernel:

$$\mathbf{K}_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \exp \left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right) = K(\mathbf{x}_i, \mathbf{x}_j), \tag{8}$$

where $\phi(.)$ represents the non-linear transformation to a reproducing kernel Hilbert space $\mathcal{H}$ [18], $\langle ., . \rangle$ denotes the inner product in $\mathcal{H}$, and $\sigma > 0$. Finally, the transformed feature vector of $\mathbf{x}$ is given by [18]:

$$\psi(\mathbf{x}) = \tilde{\Lambda}^{0.5} \mathbf{U}^\top \left[K(\mathbf{x}_1, \mathbf{x}), ..., K(\mathbf{x}_N, \mathbf{x}) \right]^\top. \tag{9}$$

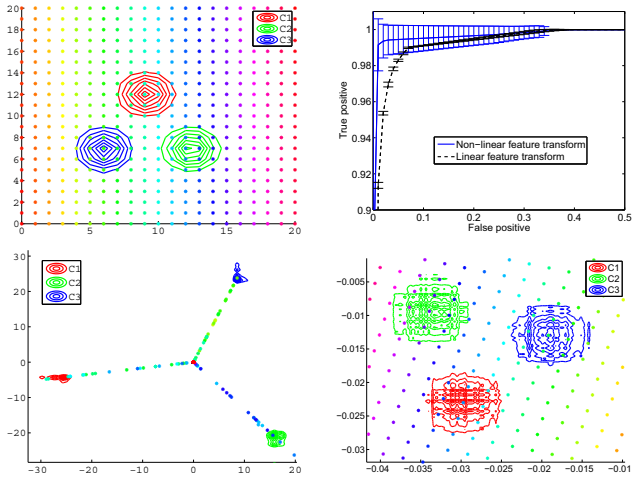


Figure 2. Distance metric learning in a 2-D data with three classes. The top-left graph represents the distributions of classes C1, C2 and C3 (contours) and several points in the original space evenly spread in a grid. The top-right graph displays the mean and standard deviation of ROC curves for the similarity-based classifier using the learned non-linear (9) and linear (4) feature transforms. The bottom-left graph shows the classes transformed using the new space produced by the non-linear distance metric learning and the transformed points from grid on the top-left graph. The bottom-right graph shows the same results for the transform produced with linear distance metric learning.

Fig. 2 displays the application of the supervised non-linear distance metric learning explained above for a three-class problem using 2-D data. Each training class (C1, C2, and C3 in the figure) is represented by a Gaussian distribution of identity covariance and respective means $\mu_1 = [9, 12], \mu_2 = [12, 7], \mu_3 = [6, 7]$. The training is processed by sampling randomly 100 points from each class and running the non-linear (9) and linear (4) feature transform learning approaches described above. This figure also shows how each point in the original space (top-left graph) is transformed in the new space (bottom graphs) according to the learned non-linear (left) and linear (right) feature transforms. Notice that in the non-linear case, the area covered by each class collapses to a small area in the transformed space, and points that are close to one of the classes in the original space converges to one of these collapsed regions. Points in the original space that falls far from any of the classes are most likely situated at the origin with a few exceptions located between the origin and one of the classes. Also notice that the maximum distance between points in the transformed space happens when points belong to different training classes, and the minimum distance occurs when points belong to the same training class. Finally, the top-right graph of Fig. 2 displays the receiver operating characteristics (ROC) curves of the classification results using a similarity-based classifier for the non-linear and linear feature transformations. To generate these curves, 10 training sets (with 100 points per class) were generated to produce the feature transforms for one test set (also with 100 points per class) using the same classes C1, C2 and C3. The ROC is built by varying the distance threshold in the transformed space and calculating the true and false positive rates. Given that we have 10 different feature transforms, we show the mean and standard deviation for the 10 ROC curves for each type of transformation. Notice how the accuracy for the non-linear transform is much bigger than that for the linear transform (*i.e.*, smaller bias), but the precision is significantly smaller (*i.e.*, bigger variance), indicating the instability mentioned before (this probably happens due to over-fitting of the training data).

2.2. Combining the Non-linear Feature Transforms

The distance between two descriptors $\mathbf{x}_i$ and $\mathbf{x}_j$ in the transformed space is computed through the following aggregation strategy:

$$D(\mathbf{x}_i, \mathbf{x}_j) = \sum_{\mathcal{K}} \|\psi_{\mathcal{K}}(\mathbf{x}_i) - \psi_{\mathcal{K}}(\mathbf{x}_j)\|_2^2, \qquad (10)$$

where $\psi_{\mathcal{K}}(.)$ is the transformation computed from the $\mathcal{K}^{th}$ training set as described in (9).

Empirically, we observe that the following conditions are necessary for our method to work: 1) the original feature values of the training points have to be in the same range as the feature values of the test points of future matching problems; and 2) the ratio of within and between training class variances has to be similar to the ratio of within and between test class variances in the original feature space. Intuitively, condition (1) increases the probability that descriptors of future matching problems have feature values different from zero in the transformed feature space. The second condition decreases the likelihood that points from the same test class are split between two training classes. Condition (2) also decreases the likelihood that points belonging to different test classes are positively classified by the same training class.

In order to explain the intuition of the functionality of our method we first define the indicator function:

$$\iota(\mathbf{x}, c) = \begin{cases} 1, & \text{if } p(\mathbf{x}|c) > \tau, \\ 0, & \text{otherwise} \end{cases}, \qquad (11)$$

to determine whether a test feature $\mathbf{x}$ is classified as belonging to training class c, which is 1 if the conditional likelihood $p(\mathbf{x}|c)$ is larger than a threshold τ, and therefore will likely be located in the collapsed region representing the training class in the transformed feature space. The distance (10) can then be rewritten as follows:

$$D(\mathbf{x}_i, \mathbf{x}_j) \approx \sum_{\mathcal{K}} \; d_1(e_1^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)) + d_2(e_2^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)) + d_3(e_3^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)) + d_4(e_4^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)), \qquad (12)$$

where the events $e_l^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)$ for each training set $\mathcal{K}$ ($e_l^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) \in \{0, 1\}$ and $\sum_{l=1}^{4} e_l^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) = 1$) and distances d_l are defined as follows:

- $e_1^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) = \prod_{c_1 \in \mathcal{K}}(1 - \iota(\mathbf{x}_i, c_1)) \prod_{c_2 \in \mathcal{K}}(1 - \iota(\mathbf{x}_j, c_2))$, representing the event that none of test descriptors fall in any of the training classes of training set $\mathcal{K}$, so both $\mathbf{x}_i$ and $\mathbf{x}_j$ collapse to a region around the origin of the new feature space;

- $e_2^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) = 1 - \prod_{c_1 \in \mathcal{K}}(1 - \iota(\mathbf{x}_i, c_1)\iota(\mathbf{x}_j, c_1))$, representing the event that both descriptors fall in the same training class of training set $\mathcal{K}$, which means that both $\mathbf{x}_i$ and $\mathbf{x}_j$ collapse to a small area covered by the training class in the transformed space;

- $e_3^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) = (1 - \prod_{c_1 \in \mathcal{K}}(1 - \iota(\mathbf{x}_i, c_1)) \prod_{c_2 \in \mathcal{K}}(1 - \iota(\mathbf{x}_j, c_2)) + \prod_{c_1 \in \mathcal{K}}(1 - \iota(\mathbf{x}_i, c_1))(1 - \prod_{c_2 \in \mathcal{K}}(1 - \iota(\mathbf{x}_j, c_2)))$, representing the event that descriptor $\mathbf{x}_i$ falls in one of the training classes of training set $\mathcal{K}$, while descriptor $\mathbf{x}_j$ does not fall in any training class of the same training set, or descriptor $\mathbf{x}_i$ does not fall in any training class of training set $\mathcal{K}$ and descriptor $\mathbf{x}_j$ falls in one of the training classes of training set $\mathcal{K}$;

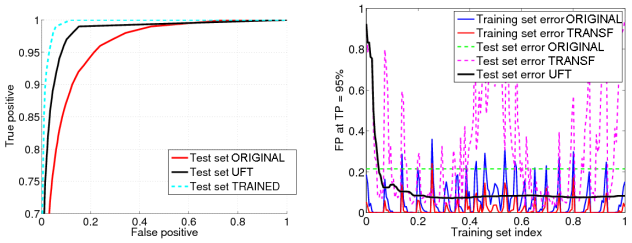
- $e_4^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j) = (1 - \prod_{c_1 \in \mathcal{K}}(1 - \iota(\mathbf{x}_i, c_1))(1 - \prod_{c_2 \in \mathcal{K}}(1 - \iota(\mathbf{x}_j, c_2))(1 - \prod_{c_3 \in \mathcal{K}} \iota(\mathbf{x}_i, c_3)\iota(\mathbf{x}_j, c_3))$, representing the event that descriptors $\mathbf{x}_i$ and $\mathbf{x}_j$ fall in two different training classes of training set $\mathcal{K}$;

- the distances d_i have the conditional expected value $E[|\|\psi(\mathbf{x}_i) - \psi(\mathbf{x}_j)\||e_i^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j))]$, and in general $d_2 < d_1 < d_3 < d_4$.

Denoting the likelihood of an event l (for $l \in \{1, 2, 3, 4\}$) between two test points $\mathbf{x}_i$ and $\mathbf{x}_j$, given that they belong to the same test class as $p(e_l^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j)$ we observe that $p(e_1^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j) > p(e_2^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j) > p(e_3^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j) > p(e_4^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j)$. On the other hand, we also note that for test points belonging to different test classes, we have: $p(e_3^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i \neq y_j) > p(e_1^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i \neq y_j) > p(e_4^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i \neq y_j) > p(e_2^{(\mathcal{K})}(\mathbf{x}_i, \mathbf{x}_j)|y_i \neq y_j)$. Therefore, if our method can substantially increase $\frac{d_4}{d_1+d_2+d_3+d_4}$ and decrease $\frac{d_2}{d_1+d_2+d_3+d_4}$ in the transformed space (compared to the original space), the average distance between test points of the same class is likely to be relatively much smaller than points from different classes when compared with the same distances in the original space.

2.3. Simple Example

In this section, we show an example that demonstrates the functionality of our approach and illustrates the intuition explained before. In order to facilitate the explanation, note that the test set represents the problem one intends to solve, and the training sets are the ones available to train the feature spaces to be combined. For the problems in this section, each point $(\mathbf{x}_i, y_i)$ with $\mathbf{x}_i \in \Re^n$ and $y_i \in \{1, 2\}$ is sampled using the following Gaussian distribution: $\mathbf{x}_i \sim G(\mu_{y_i}, \sigma_{y_i}^2)$. The set of training classes is represented by $\mathcal{R}^{\text{train}}$, and the set of test classes is denoted by $\mathcal{R}^{\text{test}}$, where $\mathcal{R}^{\text{test}} \bigcap \mathcal{R}^{\text{train}} = \emptyset$.



(a) (b)

Figure 3. Graph (a) displays the ROC curves of the test set in the original feature space (red curve) and transformed (UFT) space (black curve). For comparison purposes, we also included the curve (labeled 'test set TRAINED', cyan dashed) for the case where the feature space described in Sec. 2.1 was trained specifically for the test classes. Graph (b) shows the evolution of the false detection rate for a true positive rate of 95% in the ROC curve. For each trained feature space (the hor. axis depicts an index to each of the 100 feature spaces trained, but note that the index $\in [0, 1]$) the training set errors in the original (blue, solid) and transformed (red, solid) feature spaces are shown. Also, the error results for the test sets in the original (green, dashed) and transformed spaces (magenta, dashed) are displayed along with the error result of the UFT feature space (black, thick solid) described in Eq. 10.

The simple example involves a 1-D problem, such that $\sigma_{y_i} = 1$, and $\mu_{y_i} \in [0, 20]$. First, retrieve two random test classes to form $\mathcal{R}^{\text{test}}$ with the restriction that $|\mu_1 - \mu_2| = 4$, and randomly generate 200 test points (100 for each class) according to the distribution for the test class. This is the test problem for which we want to create a feature space that provides a more accurate threshold-based matching classifier. Then, randomly select a certain number of sets $\mathcal{K} \in \mathcal{R}^{\text{train}}$, where each set contains two training classes with $|\mu_1 - \mu_2| \approx 4$. For these experiments, we select 100 sets $\mathcal{K}$, each used to produce one feature space. Notice that this setup respects all the assumptions (same range of training and test points, and similar within and between training and test class variances) made in Sec. 2.2. For each training set $\mathcal{K} \in \mathcal{R}^{\text{train}}$, we train the non-linear feature transform described in Sec. 2.1 for 200 training points (100 for each class) randomly generated according to the distribution of the respective classes. Then given all trained feature spaces, the distance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is computed with the aggregation approach (10). Fig. 3-(a) displays the ROC curve comparing the results of the of the aggregated feature space (UFT) with the results in the original space and the results of a non-linear feature space (9) trained specifically for that test set. Fig. 3-(b) depicts the value of the false positive rate (from the ROC curve) at the operating point where the true positive rate is 95% (same criterion proposed by Winder and Brown [22]). In this graph, the following curves of the evolution of the false positive error rates are shown: the training set errors in the original and transformed spaces, the test set errors in the original and transformed spaces for each feature space, and the UFT

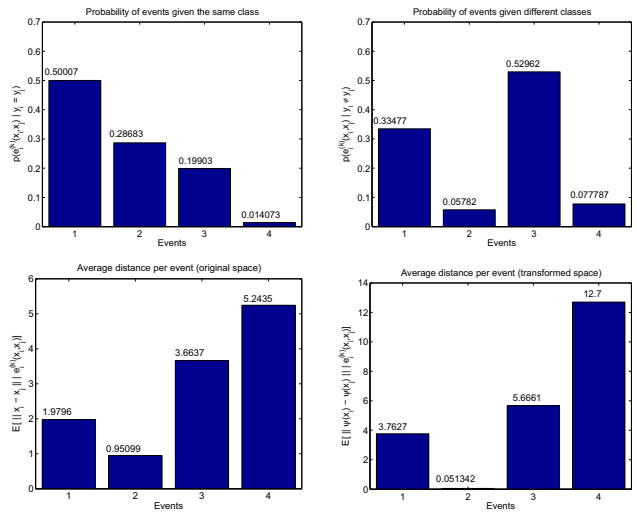


Figure 4. The first row shows the probability of each one of the four events given pair of test points from the same class (right graph) and from different classes (left). The second row displays the average distances in the original (right) and transformed (left) space given each of the four events.

error.

In Fig 4, we show the probability of each of the four events given test points belonging to the same and different classes (top row) and the average distances per event (bottom row) described in Sec. 2.2, where $\tau = 2\times$ the standard deviation of the training class distribution (recall that τ represents the threshold in the definition of function $\iota(\mathbf{x}, c)$ (11), which describes if a point $\mathbf{x}$ belongs to a class c). Notice that this example illustrates the points raised in that section about the event probabilities and distance modifications from original to transformed feature space.

3. Experiments

We applied the UFT on two publicly available databases built to compare the performance of local image descriptors. We first train the non-linear feature transforms using the training dataset proposed by Winder and Brown [22] (see Fig. 5). This database consists of more than $100,000$ image patches, sampled by back-projecting 3-D points to 2-D images from scene reconstructions, where each patch is labeled according to the scene location it belongs (notice that a label represents a 3-D scene location, and that each label contains between 2 and 50 patches). The changes present in each patch are due to variations in viewpoints, scene brightness and partial occlusions, but note that all patches are aligned to the same scale, orientation and position to a 64×64-pixel image patch. Given that typical interest point detectors (used to select locations, orientation and scales to extract local descriptors) have a much poorer precision than the reconstruction proposed [22], we also apply random deformations by artificially warping the patches of both the training and test sets, which introduces robustness to those deformations. Specifically, we use the following deformation values proposed by Hua *et al*. [11]: deviation of 0.25 pixels in position, 11 degrees in orientation and 12% in scale. In the experiments, we used the multi-way matches in the Trevi Fountain and Yosemite Valley data set as the training patches. The training was based on randomly selected N_{labels} labels from these datasets for training and N_{labels} different labels for validation, which were used to train $N_{\text{feature spaces}}$ non-linear feature extractors as described in Sec. 2.1. The aggregation of the feature extractors is done as described in Sec. 2.2, producing the UFT. For testing, we used the patches produced by the Notre Dame matches, from where 50,000 match pairs and 50,000 non-match pairs were randomly selected.

Figure 5. Example of the training patches [22].

All image patches $\mathbf{x}_{in} \in \Re^{64\times 64}$ (assumed to contain the gray values of the image patch) are pre-processed as follows:

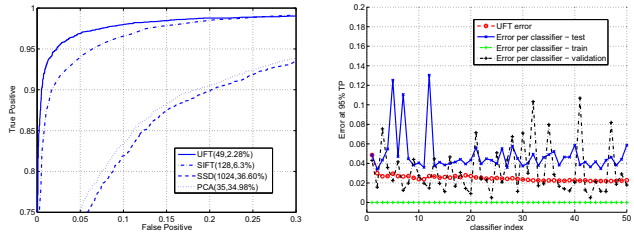
1. Normalization: $\mathbf{x}^{(1)} = \frac{\mathbf{x}_{in} - \mu_{\mathbf{x}_{in}}}{\sigma_{\mathbf{x}_{in}}}$, where $\mu_{\mathbf{x}_{in}}$ is the mean gray value of the patch and $\sigma_{\mathbf{x}_{in}}$ denotes standard deviation of the gray values of the patch;

2. Smoothing: $\mathbf{x}^{(2)} = G(0, \sigma_s) * \mathbf{x}^{(1)}$, which is a convolution between the image patch and a Gaussian filter $G(.)$ with standard deviation σ_s;

3. Spatial weighting: $\mathbf{x}^{(3)} = \mathbf{x}_w \bullet \mathbf{x}^{(2)}$, where $\mathbf{x}_w \in \Re^{64\times 64}$ is a patch containing a Gaussian centered at position $[32.5, 32.5]$ with standard deviation σ_w and $\bullet$ denotes the element-wise matrix multiplication (this operator increases the weight of points at the center of the window compared to points at the borders). The patch $\mathbf{x}^{(3)}$ is the one used for training and testing.

Using the error rate at 95% detection rate (determined with the ROC curve) over the validation set to compare performance, we reached the following values for each of the parameters above: number of training classes per feature transform $N_{\text{labels}} = 50$; number of feature transforms to build UFT $N_{\text{feature spaces}} = 50$; (although, we note that after 10 feature spaces, the performance is already stable), $\sigma_s = 2.0$ for the smoothing pre-processing, but the error results at 95% were reasonably stable for $\sigma_s \in [1.5, 4.0]$; $\sigma_w = 24$ for the spatial weighting pre-processing; and $\sigma \in [1, 20]$ in the kernel (8) is determined through cross-validation for each new feature space. Finally, using the feature learning algorithm of Sec. 2.1, the dimensionality of each transformed feature space was cross-validated to have 49 dimensions.

Fig. 6-(a) shows the ROC curve of UFT along with the curves produced by the raw patch (blurred and down-sampled to 32×32 [11]; see label SSD, standing for sum of squared distances) in its original space and its principal



(a) (b)

Figure 6. The ROC curves in (a) displays a comparison between UFT and other methods described in the text. The error rates at detection rates of 95% in graph (b) shows the evolution of the test error as new learned non-linear feature spaces are aggregated to UFT.

component analysis (PCA) space. We also show the curve produce by SIFT (designed by Vedaldi [1]) using the same parameters as by Winder *et al*. [11, 22]. We observe that the results are quite similar to those obtained by those authors, which are an error rate (at detection rate of 95%) for SIFT of 6.3%, for raw patch of 36.6% and for PCA of 34.98%. The UFT achieved an error of 2.28%, which is at the same level of the best results obtained by Hua *et al*. [11], but note that while our results were obtained from simple pre-processed image patches, the best results of [11] were obtained through a series of pre-determined image feature transforms, where the parameters of the transformations are automatically learned . If one considers the use of the original image patch gray values only (with simple pre-processing steps as the ones proposed above), then our results are substantially better than the ones shown by Hua *et al*. [11], which are around 5% to 6% at 95% detection rate. There are two important advantages with the use of simple pre-processed image patches (as the ones used in this paper): 1) potentially efficient implementation of the transformation (described below), and 2) increase of the subset of potential matching problems since we do not limit the type of input image features, as in [11]. Fig. 6-(b) displays the evolution of the error rates (again, at detection rate of 95%) by aggregating each newly learned non-linear feature transform.

Finally, we take the UFT learned above and apply to the matching problems proposed by Mikolajczyk and Schmid [15]. Using the detection methods and performance evaluation provided by the authors, we compare the performance of UFT, SIFT [14], GLOH [15] and cross-correlation (CC), using interest points detected with the Hessian-Affine detector. In order to generate the patches $\mathbf{x}_{in} \in \Re^{64\times 64}$ to be pre-processed and further transformed by UFT, we take the parameters produced by the Hessian-Affine detector, which are the position, orientation, and two dimensional scale deformation (one in the main orientation and another in an orthogonal orientation), and align the patch accordingly. We used the threshold-based matching strategy (as already mentioned above), where two regions are matched if the distance between their descriptors is below a threshold, and the image regions are considered to be a a correspondence if there is at least a 50% overlap between the regions projected onto the same image [15]. For all eight cases available on-line [15][1], UFT clearly outperforms the best features in the *1-precision versus recall* curves (see Fig. 7), which are GLOH and SIFT. The only case where the performance is comparable is for a few cases in the 'bark' sequence (first case on the top left of Fig. 7). As a reminder, these curves are computed by varying the matching threshold and calculating the following values:

$$\begin{aligned} recall &= \frac{\#\text{correct matches}}{\#\text{correspondences}}, \\ 1-precision &= \frac{\#\text{false matches}}{\#\text{correct matches}+\#\text{false matches}}, \end{aligned} \tag{13}$$

where #correct matches represents the number of correspondences having a similarity value bigger than the matching threshold, while #false matches represents the number of times a similarity bigger than the matching threshold is found in any matching (note that false matchings cannot be correspondences).

For the complexity analysis, the pre-processing part is negligible and the main run-time complexity is derived from the distance computation between test points and training points to produce the kernel matrix $\mathbf{K}_{ij}$ in (8). In general, the number of training points for each classifier is between 1000 to 2000 and the number of descriptors extracted from a test image is around 1000, so the matrix $\mathbf{K}_{ij}$ would have size $O(10^3 \times 10^3)$. We believe that with the use of methods for fast similarity computation (resorting to approximate similarity computations, for example) each feature space can be computed in negligible time. The problem of having several feature spaces is an issue that can be solved with parallel computation since each feature space is independent of all others. Therefore, the whole transformation can be computed quite fast if implemented with parallel algorithms and using approximate similarity computations.

4. Discussion and Conclusions

In this paper we proposed a new feature transform, which we call the universal feature transform, consisting of an ensemble of non-linear feature transforms. We show that it has competitive detection results for the problem of matching local image descriptors in typical image matching and classification problems. We also show empirical results demonstrating the functionality of the method and the intuition of why it works. We are currently working on a formal proof of convergence of the method based on the empirical evidence presented in this paper. We are also working on showing the applicability of this method in current matching and

[1]The bark sequence (row 1, column 1) represents zoom and rotation changes of a textured scene, the bike sequence (row 1, column 2) represents image blur deformations of a structured scene, the boat sequence (row 1, column 3) represents zoom and rotation changes of a structured scene, the graffiti scene (row 1, column 4) represents a viewpoint change of a structured scene, the Leuven sequence (row 2, column 1) represents light changes of a structured scene, the tree sequence (row 2, column 2) represents blur deformations of a textured scene, the UBC sequence (row 2, column 3) represents JPEG compression deformations, and the wall sequence (row 2, column 4) represents viewpoint changes of a textured scene

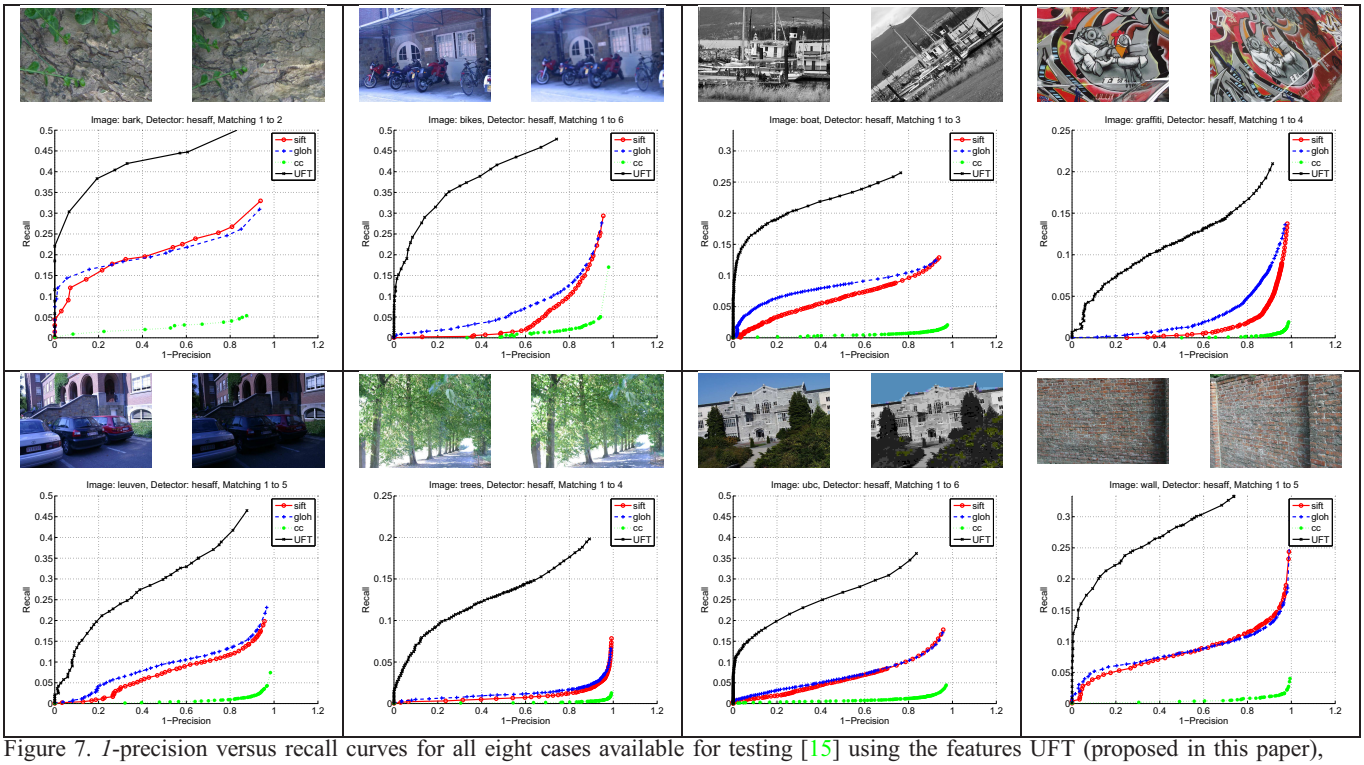


Figure 7. *1*-precision versus recall curves for all eight cases available for testing [15] using the features UFT (proposed in this paper), SIFT [14], GLOH [15] and CC, which is the raw patch cross correlation.

visual classification problems. Also, we plan to use UFT in the features designed by Hua et al. [11] in order to improve even more the performance shown by the most accurate descriptors proposed by them. Finally, we also plan to study the impact of different kernels on the results as proposed by Cristianini *et al*. [7].

References

[1] http://www.cs.ucla.edu/~vedaldi/. 7

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE TPAMI*, 24(24):509–522, 2002. 1

[3] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. 2, 3

[4] G. Carneiro and A. Jepson. Flexible spatial configuration of local image features. *IEEE TPAMI*, 29(12):2089–2104, December 2007. 2

[5] H. Chen, H. Chang, and T. Liu. Local discriminant embedding and its variants. In *CVPR*, 2005. 3

[6] S. Chopra, R. Hadsell, and Y.LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 2, 3

[7] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor On kernel-target alignment In *NIPS*, 2002. 8

[8] L. Florack, B. ter Haar Romeny, J. Koenderink, and M. Viergever. General intensity transformations and second order invariants. In *SCIA*, pages 338–345, 1991. 1

[9] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighborhood component analysis. In *NIPS*, 2004. 3

[10] X. He and P. Niyogi. Locality preserving projections. In *NIPS 16*, 2004. 3

[11] G. Hua, M. Brown, and S. Winder. Discriminant embedding for local image descriptors. In *ICCV*, 2007. 2, 6, 7, 8

[12] Y. Ke and R. Sukthankar. Pca-sift: a more distinctive representation for local image descriptors. In *CVPR*, pages 506–513, 2004. 2

[13] J. Koenderink and A. van Doorn. Representation of local geometry in the visual system. *Biological Cybernetics*, 55:367–375, 1987. 1

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 7, 8

[15] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE TPAMI*, 27(10):1615–1630, 2005. 2, 3, 7, 8

[16] E. Mortensen, H. Deng, and L. Shapiro. A sift descriptor with global context. In *CVPR*, 2005. 2

[17] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE TPAMI*, 19(5):530–535, 1997. 1

[18] M. Sugiyama. Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. *JMLR*, 8:1027–1062, 2007. 3

[19] V. N. Vapnik. *Statistical Learning Theory.* Wiley, 1998. 3

[20] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *CVPR*, 2007. 1, 2

[21] K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009. 3

[22] S. Winder and M. Brown. Learning local image descriptors. In *CVPR*, 2007. 2, 5, 6, 7