Technical Report
LDAHash: Improved matching with smaller descriptors

Christoph Strecha [*], Alexander M. Bronstein [†]
Michael M. Bronstein [‡] and Pascal Fua [§]

August 27, 2010

Abstract

SIFT-like local feature descriptors are ubiquitously employed in such computer vision applications as content-based retrieval, video analysis, copy detection, object recognition, photo-tourism and 3D reconstruction. Feature descriptors can be designed to be invariant to certain classes of photometric and geometric transformations, in particular, affine and intensity scale transformations. However, real transformations that an image can undergo can only be approximately modeled in this way, and thus most descriptors are only approximately invariant in practice. Secondly, descriptors are usually high-dimensional (e.g. SIFT is represented as a 128-dimensional vector). In large-scale retrieval and matching problems, this can pose challenges in storing and retrieving descriptor data. We map the descriptor vectors into the Hamming space, in which the Hamming metric is used to compare the resulting representations. This way, we reduce the size of the descriptors by representing them as short binary strings and learn descriptor invariance from examples. We show extensive experimental validation, demonstrating the advantage of the proposed approach.

[*]CVlab, EPFL Switzerland
[†]Tel-Aviv University, Israel
[‡]Israel Institute of Technology
[§]CVlab, EPFL Switzerland

1 Introduction

Over the last decade, feature point descriptors such as SIFT [1] and similar methods [2, 3, 4] have become indispensable tools in the computer vision community. They are usually represented as high-dimensional vectors, such as the 128-dimensional SIFT or the 64-dimensional SURF vectors. While the descriptor's high dimensionality is not an issue when only a few hundreds points need to be represented, it becomes a significant concern when millions have to be on a device with limited computational and storage resources. This happens, for example, when storing all descriptors for a large-scale urban scene on a mobile phone for image-based location purposes. Not only does this require tremendous amounts of storage, it also is slow and potentially unreliable because most recognition algorithms rely on nearest neighbor computations and computing Euclidean distances between long vectors is neither cheap nor optimal.

Consequently, there have been many recent attempts at compacting SIFT-like descriptors to allow for faster matching while retaining their outstanding recognition rates. One class of techniques relies on quantization [5, 6] and dimensionality reduction [7, 8]. While helpful, this approach does is usually not sufficient to produce truly short descriptors without loss of matching performance. Another class [9, 10, 11, 12] takes advantage of training data to learn short binary codes whose distances are small for positive training pairs and large for others. This is particularly promising because not only does binarization reduce the descriptor size, but partly also

increases performance as will be shown.

Binarization is usually performed by multiplying the descriptors by a projection matrix, subtracting a threshold vector, and retaining only the sign of the result. This maps the data into a space of binary strings, greatly reducing their size on the one hand, and simplifying their similarity computation (now becoming the Hamming metric, which can be computed very efficiently on modern CPUs) on the other.

The matrix entries and thresholds are selected so as to preserve similarity relationships in a training set. Doing this efficiently involves solving a difficult non-linear optimization problem and current methods offer no guarantee of finding a global optimum. In fact, the most successful one to date [13] is only weakly supervised, which indicates that others do not take full advantage of the training data.

To better take advantage of training data composed of interest point descriptors corresponding to multiple 3D points seen under different views, we introduce a *global* optimization scheme that is inspired by an earlier *local* optimization one [10]. In [10], the entries of the projection matrix and thresholds vectors are constructed progressively using AdaBoost. Given that Adaboost is a gradient-based method [14] and that the algorithm optimizes a few matrix rows at a time, there is no guarantee the solution it finds is optimal. By contrast, we first compute a projection matrix that is designed either to solely minimize the in-class covariance of the descriptors or to jointly minimize the in-class covariance and maximize the covariance across classes, both of which can be achieved in closed-form or using an iterative algorithm. This being done, we compute optimal thresholds that turn the projections into binary vectors so as to maximize recognition rates. In essence, we perform Linear Discriminant Analysis (LDA) on the descriptors before binarization and will therefore refer to our approach as *LDAHash*.

We use ROC curves to show that, in many different cases, using our approach to binarize SIFT descriptors [1] actually improves matching performance. This is especially true in the low false positive range with 64 or 128-bits descriptors, which means that they are about ten to twenty times shorter than the original ones. This is crucial for large-scale applications that involve matching keypoints against databases containing millions of them, which must be performed in the low false positive range to prevent list of potential matches from becoming unacceptably long. Furthermore, using competing approaches [10, 13, 15] to produce descriptors of the same size as ours results in lower matching performance over the full false positive range.

In the following section, we briefly survey existing approaches to binarization. In Section 3, we introduce our own framework. In Section 4, we describe the corresponding training methodology, training data and analyze the impact of individual components of our approach. Finally, we present our results in Section 5.

2 Prior Work

Most approaches for compacting SIFT-like descriptors and allowing for faster matching rely on one or more of the following techniques:

Tuning. In [8, 16, 6, 17, 15], the authors use training to optimize the filtering and normalization steps that produce a SIFT-like vector. The same authors optimize in [15] over the position of the elements that make up a DAISY descriptor [4].

Quantization. The SIFT descriptor can be quantized using for instance only 4 bits per coordinate [5, 15], thus saving memory and speeding up matching because comparing short vectors is faster than comparing long ones.

Dimensionality reduction. PCA has been extensively used to reduce the dimensionality of SIFT vectors [18, 6]. In this way, the number of bits required to describe each dimension can be reduced without loss in matching performance [6, 15]. In [19], a whitening linear transform was proposed in addition to benefit from the efficiency of fast nearest-neighbor search methods.

The three approaches above are mostly unsupervised methods and sometimes require a complex optimization scheme [17, 15]. Often they are not specifically tuned for keypoint matching and do not usually produce descriptors as short as one would require for large scale keypoint matching.

Our formulation relates to supervised metric learning approaches. The problem of optimizing SIFT-like descriptors can be approached from the perspective of metric learning, where many efficient approaches have been recently developed for learning similarity between data from a training set of similar and dissimilar pairs. In particular, *similarity-sensitive hashing* (SSH) or *locality-sensitive hashing* (LSH) [9, 10, 13, 11, 12] algorithms seek to find an efficient binary representation of high-dimensional data maintaining their similarity in the new space. These methods have also been applied to *global* image descriptors and bag-of-feature representations in content-based image search [20, 21], video copy detection [22], and shape retrieval [23]. In [24, 25], Hamming embedding was used to replace vector quantization in bag-of-feature construction.

There are a few appealing properties of similarity-sensitive hashing methods in large-scale descriptor matching applications. First, such methods combine the effects of dimensionality reduction and binarization, which make the descriptors more compact and easier to store. Second, the metric between the binarized descriptors is learned from examples and renders more correctly their similarity. In particular, it is possible to take advantage of feature point redundancy and transitive closures (see Fig. 2) in the training set. Finally, comparison of binary descriptors is computationally very efficient and is amenable for efficient indexing.

Existing methods for similarity-sensitive hashing have a few serious drawbacks in our application. The method of Shakhnarovich *et al.* [10] poses the similarity-sensitive hashing problem as boosted classification and tries to find its solution by means of a standard AdaBoost algorithm. However, given that AdaBoost is a greedy algorithm equivalent to a gradient-based method [14], there is no guarantee of global optimality of the solution. The spectral hashing algorithm [13], on the other hand, has a tacit underlying assumption of Euclidean descriptor similarity, which is typically far from being correct. Moreover, it is worthwhile mentioning that spectral hashing, similarity-sensitive hashing and similar methods have so far proved to be very efficient in *retrieval* applications, in which one typically tries to achieve high recall in order not to miss a few relevant matches. The operating point in these application is at *low false negative* rates. In large-scale descriptor matching, on the other hand, one requires a *low false positive* rate. As we show in the following, existing algorithms show poor performance at this operating point.

3 Approach

Let us assume we are given a large set of keypoint descriptors. They are grouped into subsets corresponding to the same 3D points and all pairs within the subsets are therefore considered as belonging to the same class. The main idea of our method is to find a mapping from the descriptor space to the Hamming space by means of an affine map followed by a sign function, such that the Hamming distance between the binarized descriptors is as close as possible to the similarity of the given data set. This can be viewed as an instance of a more general problem of metric learning, where the target Hamming metric is parametrized by the affine mapping (projection matrix and threshold vector), and the original metric is a binary relation (similar/dissimilar descriptor class) on the training set. Alternatively, thinking of our approach as an instance of similarity-preserving hashing, we are building a hash function on the descriptor space, in which *collision probability* is related to the similarity on the training data (similar descriptors have high probability of being mapped to the same binary code). Our method involves two key steps:

1. *Projection selection.* We compute a projection matrix that is designed either to solely minimize the in-class covariance of the descriptors or to jointly minimize the in-class covariance and maximize the covariance across classes, both of which can be done in closed-form (Sec. 3.5.1 and 3.5.2). Another approach is to select a projection that maximizes the margin between positive and negative pairs of descriptors and for which we propose a global optimization scheme in Sec. 3.5.3.

2. *Threshold selection.* We find thresholds that can be used to binarize the projections so that the resulting binary strings maximize recognition rates. We show that this threshold selection is a separable problem that can be solved using one-dimensional search.

In the remainder of this section, we formalize these steps and describe them in more details.

3.1 Problem formulation

Our set of keypoint descriptors are represented as n-dimensional vectors in $\mathbb{R}^n$. We attempt to find their representation in some metric space $(\mathbb{Z}, d_{\mathbb{Z}})$ by means of a map of the form $\mathbf{y} : \mathbb{R}^n \to (\mathbb{Z}, d_{\mathbb{Z}})$. The metric $d_{\mathbb{Z}} \circ (\mathbf{y} \times \mathbf{y})$ parametrizes the similarity between the feature descriptors, which may be difficult to compute in the original representation. Our goal in finding such a mapping is two-fold. First, $\mathbb{Z}$ should be an efficient representation. This implies that $\mathbf{y}(\mathbf{x})$ requires significantly less storage than $\mathbf{x}$, and that $d_{\mathbb{Z}}(\mathbf{y}(\mathbf{x}), \mathbf{y}(\mathbf{x}'))$ is much easier to compute than, e.g., $\|\mathbf{x} - \mathbf{x}'\|$. Secondly, the metric $d_{\mathbb{Z}} \circ (\mathbf{y} \times \mathbf{y})$ should better represent some ideal descriptor similarity, in the following sense: Given a set $\mathcal{P}$ of pairs of descriptors from corresponding points in different images, e.g. the same object under different view point (referred to as *positives*) and a set $\mathcal{N}$ of pairs of descriptors from different points (*negatives*), we would like $d_{\mathbb{Z}}(\mathbf{y}(\mathbf{x}), \mathbf{y}(\mathbf{x}')) < R$ for all $(\mathbf{x}, \mathbf{x}') \in \mathcal{P}$ and $d_{\mathbb{Z}}(\mathbf{y}(\mathbf{x}), \mathbf{y}(\mathbf{x}')) > R$ for all $(\mathbf{x}, \mathbf{x}') \in \mathcal{N}$ to hold with high probability for some range R.

Setting $\mathbb{Z}$ to be the m-dimensional Hamming space $\mathbb{H}^m = \{\pm 1\}^m$, the embedding of a descriptor $\mathbf{x}$ can be expressed as an m-dimensional binary string. Here, we limit our attention to affine embeddings of the form

$$\mathbf{y} = \operatorname{sign}(\mathbf{P}\mathbf{x} + \mathbf{t}) \ , \tag{1}$$

where $\mathbf{P}$ is an $m \times n$ matrix and $\mathbf{t}$ is an $m \times 1$ vector; embeddings having more complicated forms can be obtained in a relatively straightforward manner by introducing kernels. Even under the optimistic assumption that real numbers can be quantized and represented by 8 bits, the size of the original descriptor is $8n$ bits, while the size of the binary representation is m bits. Thus, setting $m \ll n$ allows to significantly alleviate the storage complexity and potentially improve descriptor indexing.

Furthermore, the descriptor dissimilarity is computed in our representation using the Hamming metric $d_{\mathbb{H}^m}(\mathbf{y}, \mathbf{y}') = \frac{m}{2} - \frac{1}{2} \sum_{i=1}^{m} \operatorname{sign}(\mathbf{y}_i \mathbf{y}'_i)$, which is done by performing a XOR operation between $\mathbf{y}$ and $\mathbf{y}'$ and counting the number of non-zero bits in the result, an operation carried out in a single instruction on modern CPU architectures (POPCNT SSE4.2).

The embedding $\mathbf{y}$ is constructed to minimize the expectation of the Hamming metric on the set positive pairs, while maximizing it on the set of negative pairs. This can be expressed as minimization of the loss function

$$L \quad = \quad \alpha \mathrm{E}\{d_{\mathbb{H}^m}(\mathbf{y}, \mathbf{y}') | \mathcal{P}\} - \mathrm{E}\{d_{\mathbb{H}^m}(\mathbf{y}, \mathbf{y}') | \mathcal{N}\}, \tag{2}$$

with respect to the projection parameters $\mathbf{P}$ and $\mathbf{t}$. Here, α is a parameter controlling the tradeoff between false positive and false negative rates (higher α correspond to lower false negative rates). In practice, the conditional expectations $\mathrm{E}\{\cdot | \mathcal{P}\}$, $\mathrm{E}\{\cdot | \mathcal{N}\}$ are replaced by averages on a training set of positive and negative pairs of descriptors, respectively.

3.2 Similarity-sensitive hashing

In [10], the computation of optimal parameters $\mathbf{P}$ and $\mathbf{t}$ was posed as a boosted binary classification problem, where $d_{\mathbb{H}}(\mathbf{y}, \mathbf{y}')$ acts as a strong binary classifier, and each dimension of the linear projection $\operatorname{sign}(\mathbf{P}_i \mathbf{x} + t_i)$ is considered a weak classifier. This way, AdaBoost can be used to find a greedy approximation of the minimizer of (2) by progressively constructing $\mathbf{P}$ and $\mathbf{t}$. At the i-th iteration, the i-th row of the matrix $\mathbf{P}$ and the i-th element of the vector $\mathbf{t}$ are found minimizing a weighted version of (2). Since the problem is non-linear, such an optimization is a challenging problem. In [10], random projection directions were used. A better method for projection selection similar to linear discriminative analysis (LDA) was proposed [22, 26]. Weights of false positive and false negative pairs are increased, and

weights of true positive and true negative pairs are decreased, using the standard AdaBoost reweighting scheme [27]. This approach can also be thought of as construction of a locality-sensitive hash (LSH) table with explicit maximization of its performance on training data.

3.3 Spectral hashing

In [13], Weiss *et al.* show that a good binary code should minimize

$$\begin{aligned} \min \int \|\mathbf{y}(\mathbf{x}) - \mathbf{y}(\mathbf{x}')\|^2 W(\mathbf{x},\mathbf{x}') p(\mathbf{x}) p(\mathbf{x}') d\mathbf{x} d\mathbf{x}' \\ \text{s.t.} \quad \int \mathbf{y}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = 0 \\ \int \mathbf{y}(\mathbf{x}) \mathbf{y}(\mathbf{x})^{\mathrm{T}} p(\mathbf{x}) d\mathbf{x} = \mathbf{I} \\ \mathbf{y}(\mathbf{x}) \in \mathbb{H}^m, \end{aligned}$$

where $W(\mathbf{x},\mathbf{x}')$ is some weight of a pair $(\mathbf{x},\mathbf{x}')$ and p is the probability density function of the descriptor vectors. The first two constraint require the bits to be zero or one with equal probability and uncorrelated between themselves, which is synonymous to code *efficiency*. Relaxing the third constraint, the problem becomes a spectral problem $L_p y = \lambda y$, whose solutions are eigenfunction of the weighted Laplacian L_p of the distribution p. Further assuming $W(\mathbf{x},\mathbf{x}') = \exp\{-\|\mathbf{x}-\mathbf{x}'\|^2/\epsilon^2\}$ and that $p(\mathbf{x}) = \prod_i u_i(x_i)$ is a separable uniform distribution with each x_i distributed on $[a_i, b_i]$, the eigenfunctions have a closed form $y(x_i) = \cos\left(\frac{k\pi x_i}{b_i - a_i}\right)$. The authors show empirically on several examples that such an approximation still gives superior results even for distributions different from uniform.

The idea of spectral hashing essentially boils down to calculating the PCA transformation of the descriptor vectors followed by the calculation of m smallest eigenfunctions of the Laplacian, the sign of which yields the binary code. While being strikingly simple and efficient, spectral hashing suffers from a bold disadvantage – the assumption that the L_2 distance is a valid distance between the descriptors, which essentially makes the approach unsupervised. Here, we show that alternative approaches making use of semi or full supervision yield significantly better results.

Both, similarity-sensitive hashing and spectral hashing will be evaluated against our approach, which is described next, in Section 5.

3.4 LDAHash

Here, we note that up to constants, problem (2) is equivalent to the minimization of

$$L = \mathrm{E}\left\{\mathbf{y}^{\mathrm{T}}\mathbf{y}'|\mathcal{N}\right\} - \alpha \mathrm{E}\left\{\mathbf{y}^{\mathrm{T}}\mathbf{y}'|\mathcal{P}\right\}, \quad (3)$$

or

$$L = \alpha \mathrm{E}\left\{\|\mathbf{y}-\mathbf{y}'\|^2|\mathcal{P}\right\} - \mathrm{E}\left\{\|\mathbf{y}-\mathbf{y}'\|^2|\mathcal{N}\right\}, \quad (4)$$

attempting to make the correlation of the binary codes as negative as possible for negative pairs and as positive as possible for positive pairs. Direct minimization of L is difficult since the terms $\mathbf{y}$ involve a non-differentiable sign non-linearity. While in principle smooth approximation is possible, the solution of the resulting non-convex problem in $(m+1)\times n$ variables is challenging, typically containing thousands of unknowns.

As an alternative, we propose to relax the problem removing the sign and minimizing a related function

$$\tilde{L} = \alpha \mathrm{E}\left\{\|\mathbf{P}\mathbf{x}-\mathbf{P}\mathbf{x}'\|^2|\mathcal{P}\right\} - \mathrm{E}\left\{\|\mathbf{P}\mathbf{x}-\mathbf{P}\mathbf{x}'\|^2|\mathcal{N}\right\}, \quad (5)$$

The above objective is independent of the affine term $\mathbf{t}$ and optimization can be performed over the projection matrix $\mathbf{P}$ only, which we further restrict to be orthogonal. Once the optimal matrix is found, we can fix it and minimize a smooth version of (4) with respect to $\mathbf{t}$.

3.5 Projection selection

Next, we describe three different approaches for computing $\mathbf{P}$, which we refer to as LDA, DIF and MAR and that we will compare in Section 4 and 5.

3.5.1 Linear Discriminant Analysis (LDA)

We start by observing that

$$\mathrm{E}\left\{\|\mathbf{P}\mathbf{x}-\mathbf{P}\mathbf{x}'\|^2|\mathcal{P}\right\} = \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{P}^{\mathrm{T}}\right\},$$

where $\mathbf{\Sigma}_{\mathcal{P}} = \mathrm{E}\left\{(\mathbf{x}-\mathbf{x}')(\mathbf{x}-\mathbf{x}')^{\mathrm{T}}|\mathcal{P}\right\}$ is the covariance matrix of the positive descriptor vector differences. This leads to

$$\tilde{L} \quad = \quad \alpha \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{P}^{\mathrm{T}}\right\} - \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{N}}\mathbf{P}^{\mathrm{T}}\right\},$$

with $\mathbf{\Sigma}_{\mathcal{N}} = \mathrm{E}\left\{(\mathbf{x}-\mathbf{x}')(\mathbf{x}-\mathbf{x}')^{\mathrm{T}}|\mathcal{N}\right\}$ being the covariance matrix of the negative descriptor vector differences.

Transforming the coordinates by pre-multiplying $\mathbf{x}$ by $\mathbf{\Sigma}_{\mathcal{N}}^{-1/2}$ turns the second term of $\tilde{L}$ into a constant for any unitary $\mathbf{P}$, leaving

$$\begin{aligned}\tilde{L} \quad &\propto \quad \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-\mathrm{T}/2}\mathbf{P}^{\mathrm{T}}\right\} \qquad (6)\\ &= \quad \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-1}\mathbf{P}^{\mathrm{T}}\right\} = \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{R}}\mathbf{P}^{\mathrm{T}}\right\},\end{aligned}$$

where $\mathbf{\Sigma}_{\mathcal{R}} = \mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-1}$ is the ratio of the positive and negative covariance matrices. Since $\mathbf{\Sigma}_{\mathcal{R}}$ is a symmetric positive semi-definite matrix, it admits the eigendecomposition $\mathbf{\Sigma}_{\mathcal{R}} = \mathbf{U}\mathbf{S}\mathbf{U}^{\mathrm{T}}$, where $\mathbf{S}$ is a non-negative diagonal matrix. An orthogonal $m \times n$ matrix $\mathbf{P}$ minimizing the trace of $\mathbf{P}\mathbf{\Sigma}_{\mathcal{R}}\mathbf{P}^{\mathrm{T}}$ is a projection onto the space spanned by the m smallest eigenvectors of $\mathbf{\Sigma}_{\mathcal{R}}$, $\tilde{L}$ is given by

$$\mathbf{P}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2} = (\mathbf{\Sigma}_{\mathcal{R}})_m^{-1/2}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2} = \tilde{\mathbf{S}}_m^{-1/2}\tilde{\mathbf{U}}^{\mathrm{T}}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2}, \quad (7)$$

where $\tilde{\mathbf{S}}$ is the $m \times m$ matrix with the smallest eigenvalues, and $\tilde{\mathbf{U}}$ is the $n \times m$ matrix with the corresponding eigenvectors (for notation brevity, we denote such a projection by $(\mathbf{\Sigma}_{\mathcal{R}})_m^{-1/2}$). This approach resembles the spirit of *linear discriminant analysis* (LDA). A similar technique has been introduced in [22] within the framework of boosted similarity learning. Note that the normalization of columns of $\mathbf{P}$ is unimportant since a sign function is applied to its output. However, we keep the normalization by the inverse square root of the variances, which makes the projected differences $\mathbf{P}(\mathbf{x}-\mathbf{x}')$ normal and white.

3.5.2 Difference of Covariances (DIF)

An alternative approach can be derived by observing that

$$\tilde{L} \quad = \quad \mathrm{tr}\left\{\mathbf{P}\mathbf{\Sigma}_{\mathcal{D}}\mathbf{P}^{\mathrm{T}}\right\},$$

where $\mathbf{\Sigma}_{\mathcal{D}} = \alpha\mathbf{\Sigma}_{\mathcal{P}} - \mathbf{\Sigma}_{\mathcal{N}}$. This yields

$$\mathbf{P} \quad = \quad (\mathbf{\Sigma}_{\mathcal{D}})_m^{-1/2}, \qquad (8)$$

where at most m smallest *negative* eigenvectors are selected. This selection of the projection matrix will be referred to as *covariance difference* and denoted by DIF. Note that it allows controlling the trade-off between false positive and negative rates through the parameter α, which is impossible in the LDA approach.

The limit $\alpha \to \infty$ is of particular interest, as it yields $\mathbf{\Sigma}_{\mathcal{D}} \propto \mathbf{\Sigma}_{\mathcal{P}}$. In this case, the negative covariance does not play any role in the training, which is equivalent to assuming that the differences of negative descriptor vectors are white Gaussian, $\mathbf{\Sigma}_{\mathcal{N}} = \mathbf{I}$. The corresponding projection matrix is given by

$$\mathbf{P} \quad = \quad (\mathbf{\Sigma}_{\mathcal{P}})_m^{-1/2}. \qquad (9)$$

The main advantage of this approach is that it allows learning the projection in a semi-supervised setting when only positive pairs are available.

In general, a fully-supervised approach is advantageous over its semi-supervised counterpart, which assumes a sometimes unrealistic unit covariance of the negative class differences. However, unlike the positive training set containing only pairs of knowingly matching descriptors, the negative set might be contaminated by positive pairs (a situation usually referred to as *label noise*). If such a contamination is significant, the semi-supervised setting is likely to perform better.

3.5.3 Maximum-margin projection (MAR)

An alternative to the methods presented above is an explicitly maximization of the margin between positive and negative training vectors. Let us assume the training set to be given in the form of triplets $(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-)$ such that $(\mathbf{x}, \mathbf{x}^+)$ and $(\mathbf{x}, \mathbf{x}^-)$ are positive and negative pairs, respectively. Given a projection $\mathbf{P}$, we define the *margin* of the triplet

$$\begin{aligned}\mu \quad &= \quad \|\mathbf{P}\mathbf{x} - \mathbf{P}\mathbf{x}^-\|^2 - \|\mathbf{P}\mathbf{x} - \mathbf{P}\mathbf{x}^+\|^2\\ &= \quad (\mathbf{x}-\mathbf{x}^-)^{\mathrm{T}}\mathbf{P}^{\mathrm{T}}\mathbf{P}(\mathbf{x}-\mathbf{x}^-)\\ &\quad\quad -(\mathbf{x}-\mathbf{x}^+)^{\mathrm{T}}\mathbf{P}^{\mathrm{T}}\mathbf{P}(\mathbf{x}-\mathbf{x}^+)\\ &= \quad \mathrm{tr}\left\{\mathbf{D}^{\mathrm{T}}\mathbf{R}\right\}, \qquad (10)\end{aligned}$$

where $\mathbf{D} = (\mathbf{x} - \mathbf{x}^-)(\mathbf{x} - \mathbf{x}^-)^{\mathrm{T}} - (\mathbf{x} - \mathbf{x}^+)(\mathbf{x} - \mathbf{x}^+)^{\mathrm{T}}$ and $\mathbf{R} = \mathbf{P}^{\mathrm{T}}\mathbf{P}$.

We would like to find an $m \times m$ matrix $\mathbf{R}$ maximizing $\mu_k = \mathrm{tr}\left\{\mathbf{D}_k^{\mathrm{T}}\mathbf{R}\right\}$ for each training triplet $k = 1, \ldots, K$. Using the *soft margin* formulation [28], the problem can be posed as

$$\begin{aligned} \min_{\mathbf{R},\boldsymbol{\xi},\rho} -\rho + c \cdot \mathbf{1}^{\mathrm{T}}\boldsymbol{\xi} \quad \text{s.t.} \quad & \mu_k + \xi_k \geq \rho \\ & \boldsymbol{\xi} \geq 0, \end{aligned} \tag{11}$$

where $\boldsymbol{\xi}$ is the $K \times 1$ vector of slack variables, and c is the margin violation penalty.

The original margin maximization problem (11) involving a matrix $\mathbf{R}$ is difficult to solve. Here, we propose to follow the linear programming boosting (LPBoost, [29]) spirit and devise an algorithm that gives an exact solution to (11). For this purpose, we represent the matrix $\mathbf{R}$ as a weighted sum of rank-one matrices. Each such rank-one matrix, $\mathbf{R}_\mathbf{p} = \mathbf{p}\mathbf{p}^{\mathrm{T}}$, is parametrized by a unit vector $\mathbf{p}$ in $\mathbb{R}^n$. This yields

$$\mathbf{R} = \sum_{\mathbf{p}\in\mathbb{S}^n} \alpha_\mathbf{p}\mathbf{R}_\mathbf{p} \tag{12}$$

where $\alpha_\mathbf{p} \geq 0$ are weights. In this representation, the margin of a training example k is given by

$$\begin{aligned} \mu_k &= \mathrm{tr}\left\{\mathbf{D}_k^{\mathrm{T}}\mathbf{R}\right\} = \sum_{\mathbf{p}\in\mathbb{S}^n} \alpha_\mathbf{p}\mathrm{tr}\left\{\mathbf{D}_k^{\mathrm{T}}\mathbf{R}_\mathbf{p}\right\} \\ &= \sum_{\mathbf{p}\in\mathbb{S}^n} \alpha_\mathbf{p}\mu_k(\mathbf{p}), \end{aligned} \tag{13}$$

where $\mu_k(\mathbf{p}) = \mathrm{tr}\left\{\mathbf{D}_k^{\mathrm{T}}, \mathbf{R}_\mathbf{p}\right\}$. This allows to rewrite problem (11) as a primal linear program

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\xi},\rho} -\rho + c \cdot \mathbf{1}^{\mathrm{T}}\boldsymbol{\xi} \quad \text{s.t.} \quad & \sum_{\mathbf{p}\in\mathbb{S}^n} \alpha_\mathbf{p}\boldsymbol{\mu}(\mathbf{p}) + \boldsymbol{\xi} \geq \rho \\ & \sum_{\mathbf{p}\in\mathbb{S}^n} \alpha_\mathbf{p} = 1 \\ & \alpha_\mathbf{p} \geq 0 \\ & \boldsymbol{\xi} \geq 0, \end{aligned} \tag{14}$$

where $\boldsymbol{\mu}(\mathbf{p}) = (\mu_1(\mathbf{p}), \ldots, \mu_K(\mathbf{p}))^{\mathrm{T}}$. The equivalent dual linear program of (14) is

$$\begin{aligned} \max_{\boldsymbol{\lambda},\gamma} \gamma \quad \text{s.t.} \quad & \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\mu}(\mathbf{p}) + \gamma \leq 0 \\ & 0 \leq \boldsymbol{\lambda} \leq c \\ & \mathbf{1}^{\mathrm{T}}\boldsymbol{\lambda} = 1 \end{aligned} \tag{15}$$

with $\boldsymbol{\lambda}$ being the $K \times 1$ vector of Lagrange multipliers of the primal (by duality, $\boldsymbol{\alpha}$ are Lagrange multipliers of the dual). Solutions of both problems (14) and (15) coincide.

Note that the "vector" $\boldsymbol{\alpha}$ in the primal and the constraints in the dual are infinitely dimensional. LPBoost frequently involves such large or infinitely-dimensional linear programs, for which the efficient Dantzig-Wolfe decomposition (sometimes known as column generation) is used [29]. Let us consider a subset of the satisfied constraints in the dual problem. For any finite subset, we can solve the linear program and thus satisfy all constraints. If we could prove that of all the constraints which we did not add to the dual problem no single constraint is violated, we would have proven that solving the restricted problem is equivalent to solving the original problem. More formally, let $\gamma^\star$ be the minimizer of a restricted instance of the problem. Then, we can formulate a *search problem* for the most violated constraint in the original problem space, namely finding

$$\begin{aligned} \mathbf{p}^\star &= \arg\max_{\mathbf{p}\in\mathbb{S}^n} \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\mu}(\mathbf{p}) \\ &= \arg\max_{\mathbf{p}\in\mathbb{S}^n} \sum_{k=1}^{K} \lambda_k \mathrm{tr}\left\{\mathbf{D}_k^{\mathrm{T}}\mathbf{p}\mathbf{p}^{\mathrm{T}}\right\} \\ &= \arg\max_{\mathbf{p}\in\mathbb{S}^n} \mathrm{tr}\left\{\mathbf{D}^{\mathrm{T}}\mathbf{p}\mathbf{p}^{\mathrm{T}}\right\} \\ &= \arg\max_{\mathbf{p}\in\mathbb{S}^n} \mathbf{p}^{\mathrm{T}}\mathbf{D}\mathbf{p}, \end{aligned} \tag{16}$$

where

$$\mathbf{D} = \sum_{k=1}^{K} \lambda_k \mathbf{D}_k. \tag{17}$$

The problem has a closed form solution with $\mathbf{p}^\star$ corresponding to the maximal eigenvector of $\mathbf{D}$.

Since the descriptors projected by $\mathbf{P}$ are further binarized, it is important to impose an orthogonality constraint on $\mathbf{P}$ (if $\mathbf{P}$ is not orthogonal, the resulting bits will be correlated reducing the efficiency of the code). This can be achieved by projecting $\mathbf{D}$ on the subspace orthogonal to that spanned by already selected vectors,

$$\tilde{\mathbf{D}} = (\mathbf{I} - \mathbf{Q}_m)^{\mathrm{T}}\mathbf{D}(\mathbf{I} - \mathbf{Q}_m)\ , \tag{18}$$

Initialization
1. Uniform weights $\boldsymbol{\lambda} \leftarrow \frac{1}{K} \cdot \mathbf{1}$
2. Zero margin $\gamma \leftarrow 0$

Iterations
for $m = 1, 2, \ldots$ **do**
3. Construct $\tilde{\mathbf{D}}$ according to (18)
4. $\mathbf{p}_m \leftarrow$ largest eigenvector of $\hat{\mathbf{D}}$
5. $\boldsymbol{\mu}_m \leftarrow (\mathbf{p}_m^{\mathrm{T}}\mathbf{D}_1\mathbf{q}_m, \ldots, \mathbf{p}_m^{\mathrm{T}}\mathbf{D}_K\mathbf{q}_m)^{\mathrm{T}}$
6. **if** $\boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\mu}_m + \gamma \leq \epsilon$ **then** break
7. $(\boldsymbol{\lambda}, \gamma) \leftarrow$ solution to (15)
8. $(\alpha_1, \ldots, \alpha_m) \leftarrow$ Lagrange multipliers of (15)

end

Output
9. $\mathbf{P} \leftarrow (\sqrt{\alpha_1}\mathbf{p}_1, \ldots, \sqrt{\alpha_m}\mathbf{p}_m)^{\mathrm{T}}$

Algorithm 1: LPBoost maximum-margin projection training algorithm.

where $\mathbf{P}_m = (\mathbf{p}_1, \ldots, \mathbf{p}_{m-1})$ is the matrix whose columns are the vectors selected so far and $\mathbf{Q}_m = \mathbf{P}_m(\mathbf{P}_m^{\mathrm{T}}\mathbf{P}_m)^{-1}\mathbf{P}_m^{\mathrm{T}}$.

Plugging the above ingredients into the standard LPBoost framework [29], we obtain Algorithm 1. The parameter ϵ serves as a *convergence threshold*. By setting $\epsilon = 0$ the obtained solution is the global minimizer of (11). In practice, a small positive value can be used to obtain a good solution with a small number of iterations. A bold advantage of LPBoost over other boosting algorithms such as AdaBoost is the fact that it is *totally corrective*, meaning that all the weights α_i are adjusted in each iteration. This is opposed to the greedy behavior of AdaBoost.

We denote the maximum-margin projection obtained using this boosting algorithm as MAR.

3.6 Threshold selection

Given the projection matrix $\mathbf{P}$ selected as described in the previous section, our next step is to minimize a smooth version of the loss function (3),

$$\begin{aligned} L &= \mathrm{E}\left\{\mathrm{sign}(\mathbf{P}\mathbf{x}+\mathbf{t})^{\mathrm{T}}\mathrm{sign}(\mathbf{P}\mathbf{x}'+\mathbf{t})|\mathcal{N}\right\} \\ &\quad -\alpha\mathrm{E}\left\{\mathrm{sign}(\mathbf{P}\mathbf{x}+\mathbf{t})^{\mathrm{T}}\mathrm{sign}(\mathbf{P}\mathbf{x}'+\mathbf{t})|\mathcal{P}\right\} \\ &= \sum_{i=1}^{m}\mathrm{E}\left\{\mathrm{sign}(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}+t_i)\mathrm{sign}(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}'+t_i)|\mathcal{N}\right\} \\ &\quad -\alpha\mathrm{E}\left\{\mathrm{sign}(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}+t_i)\mathrm{sign}(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}'+t_i)|\mathcal{P}\right\}, \end{aligned} \tag{19}$$

with respect to the thresholds $\mathbf{t}$, where $\mathbf{p}_i^{\mathrm{T}}$ denotes the i-th row of $\mathbf{P}$, and t_i denotes the i-th element of $\mathbf{t}$. Observe that due to its separable form, the problem can be split into independent sub-problems

$$\begin{aligned} \min_{t_i} \quad & \mathrm{E}\left\{\mathrm{sign}((\mathbf{p}_i^{\mathrm{T}}\mathbf{x}+t_i)(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}'+t_i))|\mathcal{N}\right\} \\ & -\alpha\mathrm{E}\left\{\mathrm{sign}((\mathbf{p}_i^{\mathrm{T}}\mathbf{x}+t_i)(\mathbf{p}_i^{\mathrm{T}}\mathbf{x}'+t_i))|\mathcal{P}\right\}, \end{aligned} \tag{20}$$

which in turn can be solved using simple one-dimensional search over each threshold t_i.

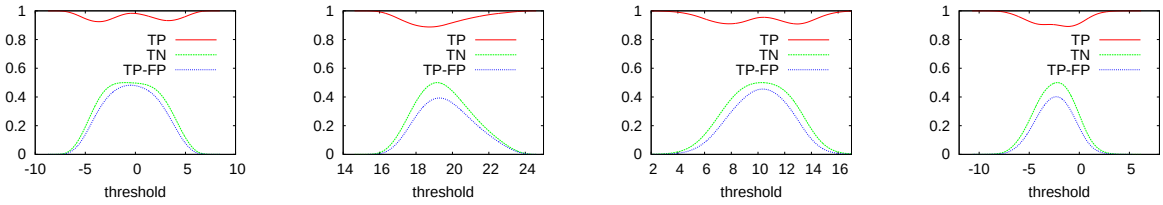
Let $y = \mathbf{p}_i^{\mathrm{T}}\mathbf{x}$ and $y' = \mathbf{p}_i^{\mathrm{T}}\mathbf{x}'$ be the i-th element of the projected training vectors $\mathbf{x}$ and $\mathbf{x}'$. The i-th bits of $\mathbf{y}$ and $\mathbf{y}'$ coincide if $t_i < \min\{y, y'\}$ or $t_i > \max\{y, y'\}$, and differ if $\min\{y, y'\} \leq t_i \leq \max\{y, y'\}$. For a given value of the threshold, we express the false negative rate as

$$\begin{aligned} \mathrm{FN}(t) &= \Pr(\min\{y, y'\} \geq t \text{ or } \max\{y, y'\} < t|\mathcal{P}) \\ &= 1 - \Pr(\min\{y, y'\} < t|\mathcal{P}) + \Pr(\max\{y, y'\} < t|\mathcal{P}) \\ &= 1 - \mathrm{cdf}(\min\{y, y'\}|\mathcal{P}) + \mathrm{cdf}(\max\{y, y'\}|\mathcal{P}). \end{aligned} \tag{21}$$

Similarly, false positive rate can be expressed as

$$\begin{aligned} \mathrm{FP}(t) &= \Pr(\min\{y, y'\} < t \leq \max\{y, y'\}|\mathcal{N}) \\ &= 1 - \Pr(\min\{y, y'\} \geq t \text{ or } \max\{y, y'\} < t|\mathcal{N}) \\ &= \mathrm{cdf}(\min\{y, y'\}|\mathcal{N}) - \mathrm{cdf}(\max\{y, y'\}|\mathcal{N}). \end{aligned} \tag{22}$$

We compute histograms of minimal and maximal values of projected positive and negative pairs, from which the cumulative densities are estimated. The optimal threshold t_i is selected to minimize $\mathrm{FP} + \mathrm{FN}$ (or, alternatively, maximize $\mathrm{TN} + \mathrm{TP}$, where $\mathrm{TP} = 1 - \mathrm{FN}$ and $\mathrm{TN} = 1 - \mathrm{FP}$ are the true positive and true negative rates, respectively). Figure 1 visualizes TP, TN and $\mathrm{TP} - \mathrm{FP}$ for the first two components $i = 1, 2$ of the projections LDA and DIF.



(a) (b) (c) (d)

Figure 1: The probability density functions for the classification performance for positive and negative training examples for the first two dimensions (a) and (b) for LDA for the first two dimensions (c) and (d) for DIF.

4 Training Methodology

In this section, we first describe our ground truth training and evaluation data. We then evaluate different aspects of our binary descriptors.

4.1 Ground Truth Data

To build our ground truth database, we used sets of calibrated images for which we show the 3-D point model and a member image in Figures 2, 3, 12, 13, 14 and 15. These datasets contain images we acquired ourself, such as those in Fig. 12 and d13), and sometimes over extended periods of time (Figs. 2). Those of Figs. 2, 3, 13, 14 are downloaded from the internet or are fully acquired from this source, as in the case of Fig. 15.

We used our own calibration pipeline [30] to register them and to compute internal and external camera parameters. Correspondences between keypoints were established using Vedaldi's [31] SIFT [1] descriptors that we compared using the standard L_2-norm.

Because our dataset contains multiple views of the same scene, we have many conjunctive closure matches [32] such as the one depicted by the blue line in Figure 2 (bottom): a keypoint that is matched in two other images, as depicted by the green lines, gives rise to an additional match in these other two images. Since they may be quite different from each other, the L_2 distance between the corresponding descriptors may be large. Yet, the descriptors in all three images will be treated as belonging to the same class, which is key to learning a metric that can achieve better matching performance than the original L_2 norm. In our datasets, these conjunctive closures partially build long chains for which individual pairs can have quite large L_2 norm as one can see in Figures 17 and 17. In practice, we consider only chains with 5 or more keypoints, *i.e.* 3-D points that are visible in at least 5 images.

For the negative examples, we randomly sampled the same number of keypoint pairs and checked that none of them belonged to the positive set.

This training database is more specific than the one used by [8, 16], where the authors use a calibrated database of images and their dense multi-view stereo correspondences. However, calibration and dense stereo information is used to extract the image patches which are centered around 3-D point projections and use these to build a training database of positive matches. In our framework, we use the calibration only to geometrically *verify* SIFT matches as being consistent with the camera parameters and with the 3-D structure. The 2-D position, scale and orientation of the original interest points is kept, such that we can perform learning on the data, that is actually extracted by the combination of SIFT keypoint detection (DoG) and description.

In [6, 15] stereo correspondences are used to build a training database of positive keypoint pairs, similar to ours. This approach has advantages if the computed stereo correspondences are reliable even for image pairs with strong appearance changes. However, it is likely that ground truth correspondences

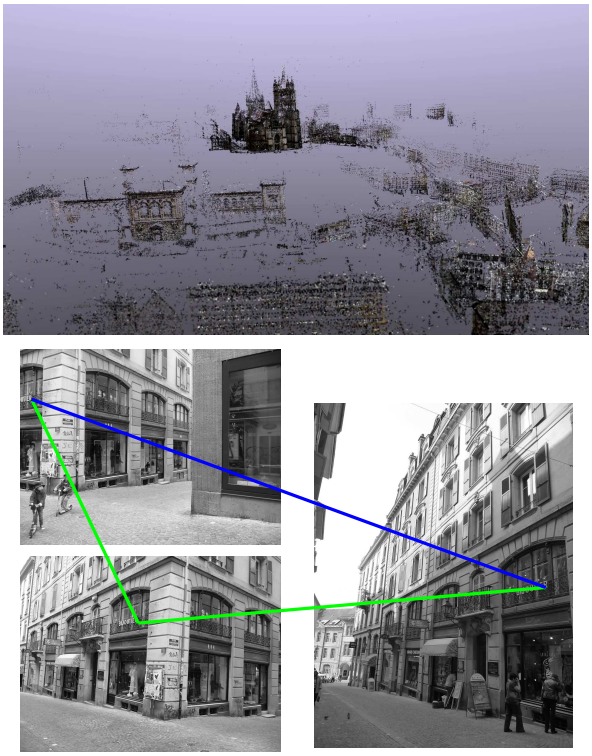


Figure 2: Top row: Calibrated model of Lausanne with 4485 images and 1.264M 3D points that are computed from 9.9M feature points. Bottom row: Three sample images from the dataset with a transitive closure indicated.

for which SIFT already give good results are over-represented by this strategy [15]. Here we put more effort to build long chains of subsequent matches, that end up describing the huge variability of features represented by the same 3-D point. Samples of these feature chains are shown in Figures 16 and 17. One can see that they cover a large variability, both in scale and appearance. Grouping those keypoints into the same class was possible by subsequently lining up small difference SIFT matches into a long chain.

To train our descriptors we use the Lausanne dataset of Figure. 2. Approximately 9.9M feature points are extracted and their triangulation produced

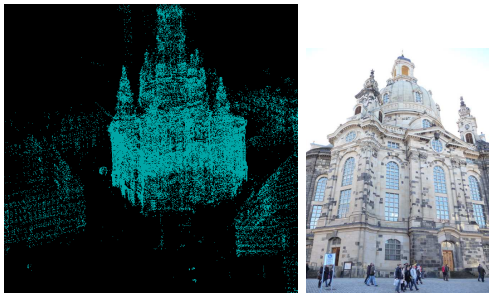


Figure 3: Dresden dataset used for the evaluation in Figures 5 and 6 contains 4.551.124 positive and negative matches, which are obtained by geometric verification using the full calibration.

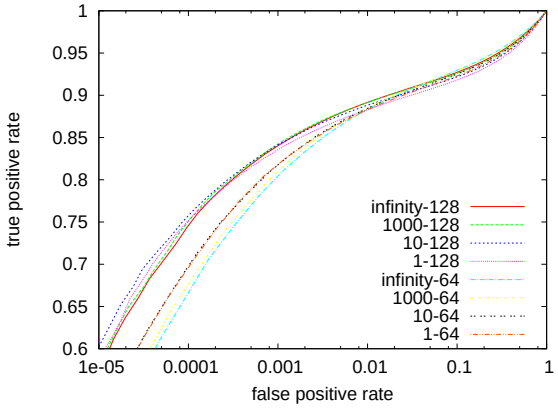


Figure 4: Performance evaluation for the DIF binarization as a function of α for 128 and 64 bits on the Dresden dataset shown in Figure 3. The label on each curve indicates $\alpha-$number of bits.

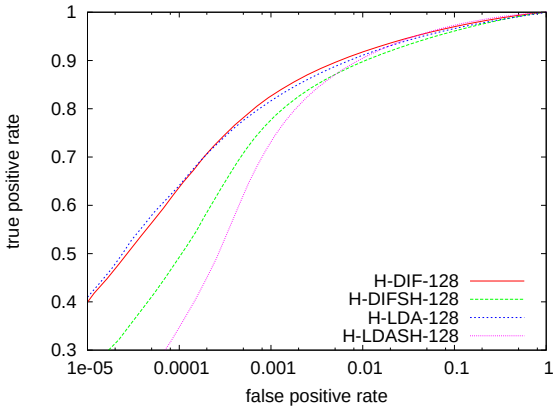


Figure 5: Performance evaluation for the binarization used in spectral hashing [13] (denoted by the ending SH for each projection) with our proposed threshold optimization in Section 3.6 for the Venice dataset shown in Figure 15. Note, that our threshold selection outperforms the corresponding SH formulation over the full false positive range.

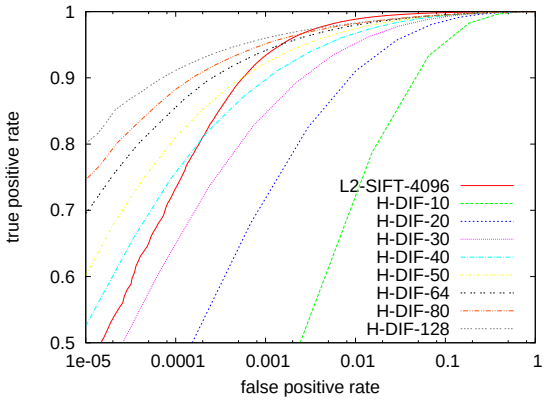


Figure 7: Performance of DIF with varying number of bits on the Karls bridge dataset of Prague (Figure 14). As a reference we include the original SIFT performance.

about 1.3M 3-D points, such as those depicted in the top of Figure 2. The urban area represented here covers nearly 2 square kilometers and encompasses the appearance statistics of man-made scenes. Vegetation also appears but is not extensively represented in this database. These training database finally consists of about 72M positive and negative matching pairs from nearly 8M keypoints. For testing we used the datasets in Figures 3, 12, 13, 14 and 15 as well as Lidar ground truth data and planar image pairs as described Section 5.1.

4.2 Parameter Evaluation

In the following we evaluate the two steps in our optimization: *i)* the computation of $\mathbf{P}$, which results in a dimensionality reduced floating point feature vector and *ii)* the estimation of the thresholds that perform the binarization. For this evaluation we use a set of images from different cities of Figures 3, 12, 13, 14 and 15. These provide positive and negative matching examples, which we use to compute the ROC statistics for different descriptor distances, i.e. L_2-ball or Hamming cube. We use the same negative samples in all cases.

All ROC curves are plotted in log scale for the FP rate, since the operating point for large scale image retrieval systems require very low FP rates. For example a value of FP $= 0.01$ (1%) for the Dresden dataset with 4.5M positive and negative matching examples will result in 45K false positives, which is far more than retrieval systems could possibly handle. We are thus interested in performance at FP $\ll 1\%$.

Throughout the paper, we use the following convention to the algorithms we compare: *Metric-Projection-Size*. The *metric* can either be L_2 (Euclidean) or H (Hamming on the binarized vectors). *Projection* denotes the way in which the projection matrix $\mathbf{P}$ is computed: LDA (linear discriminant according to equation (7)), DIF (difference of covariances according to equation (8) or MAR (maximal margin projection according to Algorithm 1). *Size* denotes the descriptor length in bits.

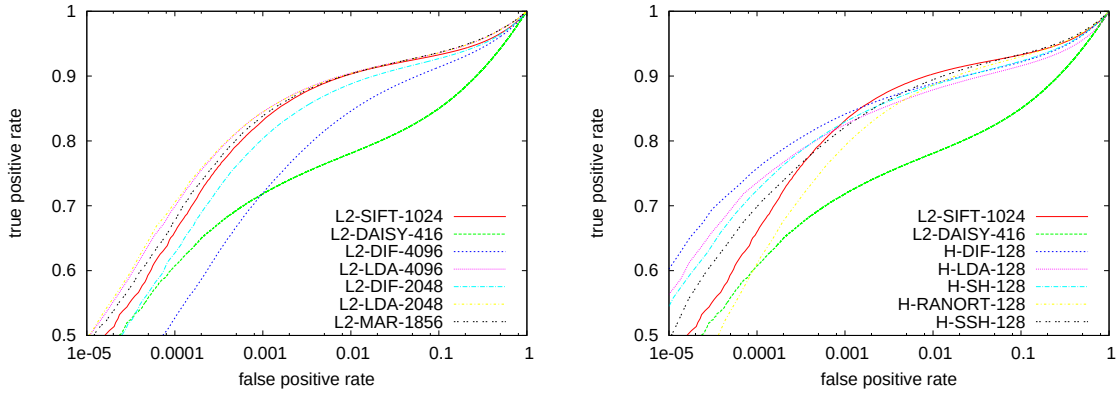


Figure 6: Left: Performance evaluation for the projection **P** for our three methods (DIF, LDA and MAR) in comparison to the original SIFT and to the DAISY descriptor on the Dresden dataset shown in Figure 3. Right: Performance evaluation for various descriptors for the same dataset after binarization. We compare our binary descriptors with Locality Sensitive Hashing by [10] (H-SSH-128), DAISY [6] (L2-DAISY-416), SIFT [1] (L2-SIFT-1024) and random orthogonal projections (H-RANORT-128). Note, that binarization improves the performance for the interesting (low false positive rate) area of the ROC curves.

4.3 The choice of α in DIF projections

Figure 4 shows the performance of the DIF formulation when the relative influence of positive and negative training data is varied. This is achieved by α in Equation 8. $\alpha = 10$ leads to the best results for both, 128 and 64 bit descriptors. Note that this experiments also includes the case where only positive matches are taken into account, *i.e.* the approach with $\alpha = \infty$. All remaining results in this paper will therefore use $\alpha = 10$ and we denote the corresponding binarization by DIF.

4.4 Linear Projection

We estimated the parameters **P** of our projection matrix of (1) to produce descriptors of size $m = 64$ and 128 for DIF and LDAand for $m = 58$ for MAR. The projection by **P** results in floating point descriptors $\mathbf{y} = \mathbf{Px}$ which we compare in Figure 6 (left) to SIFT [1] [31] and to DAISY [6, 15]. For DAISY, we used software provided by Simon Winder, who also suggested the optimal parameters.[1]

As shown in Figure 6 (left) LDA and MAR projections improve the results when compared to SIFT. By contrast, DIF projections performs worse than the original SIFT descriptors. This effect is stronger when we reduce the dimensions to 64. However, after binarization, these results change as will be shown next.

4.5 Binarization

In Figure 5, we compare our supervised threshold optimization with the spectral hashing approach [13], which has been shown to outperform many other hashing approaches such as restricted Boltzmann machines and locality-sensitive hashing [13]. Spectral hashing first applies a PCA projection of the feature space. Then the bounding box of all feature vectors is computed and the binarization is realized by

[1] The DAISY parameters used: *i)* the keypoint scale, which transforms the SIFT scale parameter to DAISY scale, was set to 1.6 and *ii)* the descriptor **T2 4 2r6s** making up a 52 dimensional feature representation of unsigned char values was used in all experiments. For additional details, see [6, 15].

looking at the sign of the analytical eigenfunctions in that box for each dimension. The SH approach selects the m smallest of those eigenfunctions. Instead of applying PCA projections, we show the performance of this particular binarization scheme for DIF and LDA projections, denoted as H-DIFSH-128 and H-LDASH-128. This is compared to our supervised threshold optimization (H-DIF-128 and H-LDA-128) in Fig. 5. One can see that our binarization scheme, as described in Section 3.6 does increase performance substantially over the corresponding spectral hashing formulation. Note also that SH binarization is related to feature discretization, which tries to approximate floating point feature vectors by fewer bits in each dimension. Without sorting the m smallest eigenfunctions, or equally scaling each dimension of the feature space to the same range, SH corresponds to a discretization of each feature dimension into several bits[2]. Un-supervised feature discretization, as used by Brown *et.al.* [15], will therefore show a similar behavior as SH binarization does.

4.6 Combined Comparison

In Figure 6 (right) we show the final result of our binarized descriptors in comparison to other approaches. One can see that if the data are transformed according to the covariance structure of the feature space (by LDA or DIF), we get a significant performance boost by using the Hamming metric on binarized descriptors. This can be seen even for H-DIF-128, for which the un-binarized version L2-DIF-4096 performs worse than SIFT. If, on the other hand, the feature space is not aligned with the covariance structure, binarization does not improve, e.g., for random orthogonal projections H-RANORTH-128. Figure 6 also shows the results of similarity-sensitive hashing proposed by [10] and used in [33], the results of DAISY [6, 15] and spectral hashing [13]. Our approach shows significantly better performance in the interesting area of low false positive rates and reaches the performance of the other descriptors for high true positive rates with a much smaller descriptor size. In the next sections (5.1 and 5.2) we show similar or a better behavior on more difficult datasets of our approach on many other test sequences.

Note also, the improvement of the binarization with respect to the un-binarized projection by comparing Figures 6 (left) and (right) for LDA and DIF. An improvement by quantization was also reported by Brown *et.al.* [15], where the range of each descriptor coordinate has been binarized to fit various bit sizes.

In Figure 7 we show the performance with varying number of bits for DIF binarization and we compare it to the SIFT baseline performance.

5 Experimental Evaluation

In this section, we compare the performance of our approach to metric learning against state-of-the-art methodss [10, 13, 15] and use SIFT [1] as a baseline. We first do this using image pairs for which LIDAR data, and therefore ground truth correspondences, are available. We then move on to the large scale datasets presented in Section 5.2 to validate our approach in a more challenging context.

5.1 LIDAR ground truth evaluation

We evaluated the performance of our binarized descriptor on publicly available datasets [35, 34], for which camera parameters and the ground truth 3D model are available. The dense ground-truth cloud of 3D points was obtained by using LIDAR and was registered to the images, making it easy to find the corresponding pixel in any image to a pixel in any other. Occluded areas can by identified, and have been excluded from the evaluation, by geometric visibility reasoning. This high precision evaluation data does contain real 3-D distortions which is different from the well know dataset of Mikolajczyk *et.al.* [2], where the images are related by a single homography. It does therefore allow to evaluate more realistic scenarios.

We focus on two pairs of the Fountain-P11 and the Herz-Jesu-P8 datasets depicted in Figure 8. For both dataset we present the results for a small baseline and

[2]The number of bits depends on the frequency of the harmonic eigenfunctions and can be chosen (see [13] for more details).

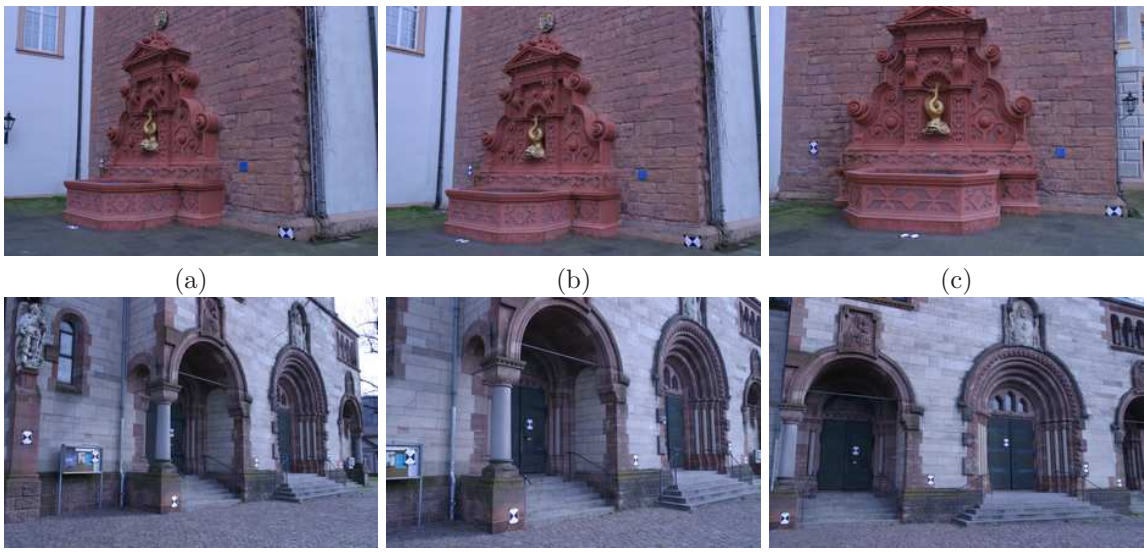
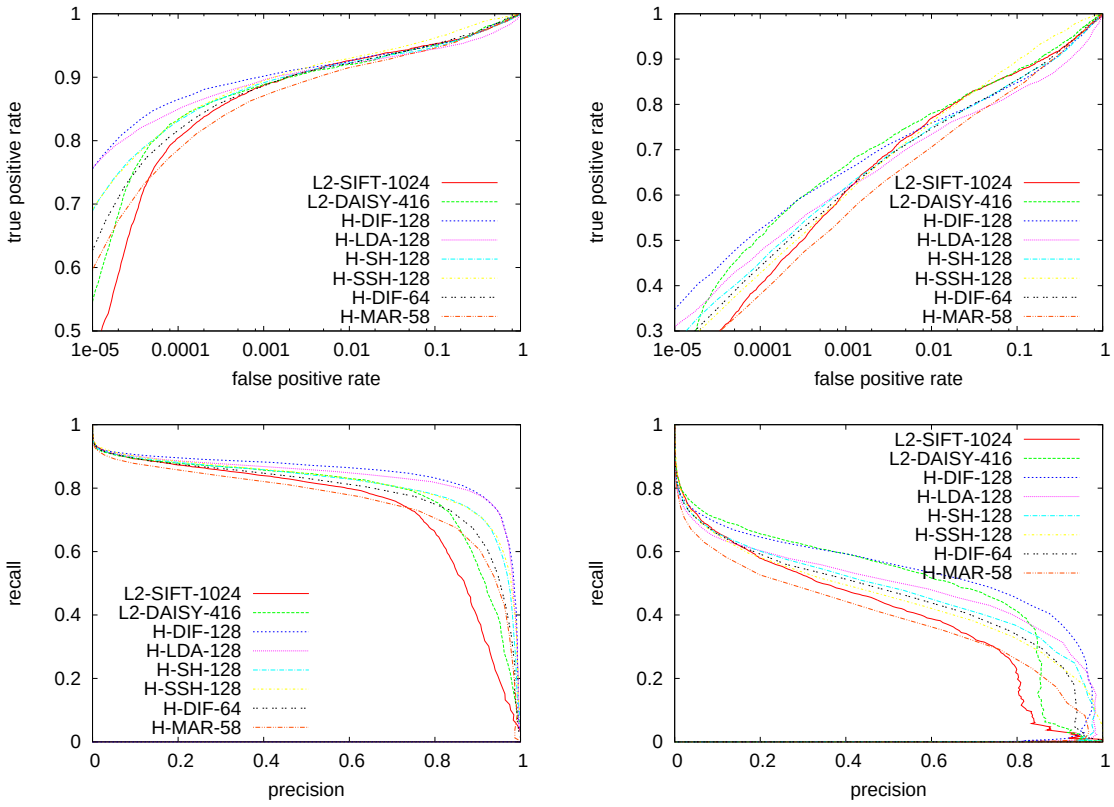


Figure 8: Images used for quantitative evaluation. Dense ground-truth correspondences are available from LIDAR measurements for fountain-P11 (top) and Herz-Jesu-P8 (bottom) [34]. The matching performance of the image pairs a-b and a-c as well as d-e and d-f are shown in Figures 9 and 10.

a wide baseline setting. These datasets and the evaluation procedure will be publically available [36]. In addition, we show results on the standard graffiti and wall datasets of Mikolajczyk *et.al.*[2], which consists of planar scenes, making it easy to establish dense correspondence by a homography. In Figures 9, 10 and 11, we plot ROC and precision-recall curves that summarize the corresponding matching performance using the various descriptors. These curves were obtained as follows: First, SIFT keypoints were detected in all images. From these, we filtered out all keypoints for which there were no ground truth matches, either due to missing LIDAR data or occlusions. For each of the remaining keypoints in one image, we search for the corresponding keypoint in the other image and check whether it is less than two pixels[3] away from the ground truth LIDAR match. To enforce consistency, we switched the roles of the images and performed the same operation. This provided us with ground truth keypoint correspondences and we further did the evaluation only on those keypoints. By varying the matching threshold on either the L_2-norm or Hamming distance, we counted the number of true and false positives to obtain the ROC curves. By using the same set of keypoints the recall is defined by the relative amount of true positives and precision by the number of true positive relative to the total number of retrieved keypoints.

In the fountain-P11 and Herz-Jesu-P8 datasets (Figures 9 and 10) the 128-bit binary descriptors significantly outperform SIFT. This performance boost is achieved with a descriptor size which is 8 times less than the number of bits original SIFT requires (1024). Even if we half the size of our descriptors to 64 bits we get results that are similar and in some cases superior to those of SIFT in accuracy, while being 16 times more compact. These experiments show a significant improvement of DAISY when compared

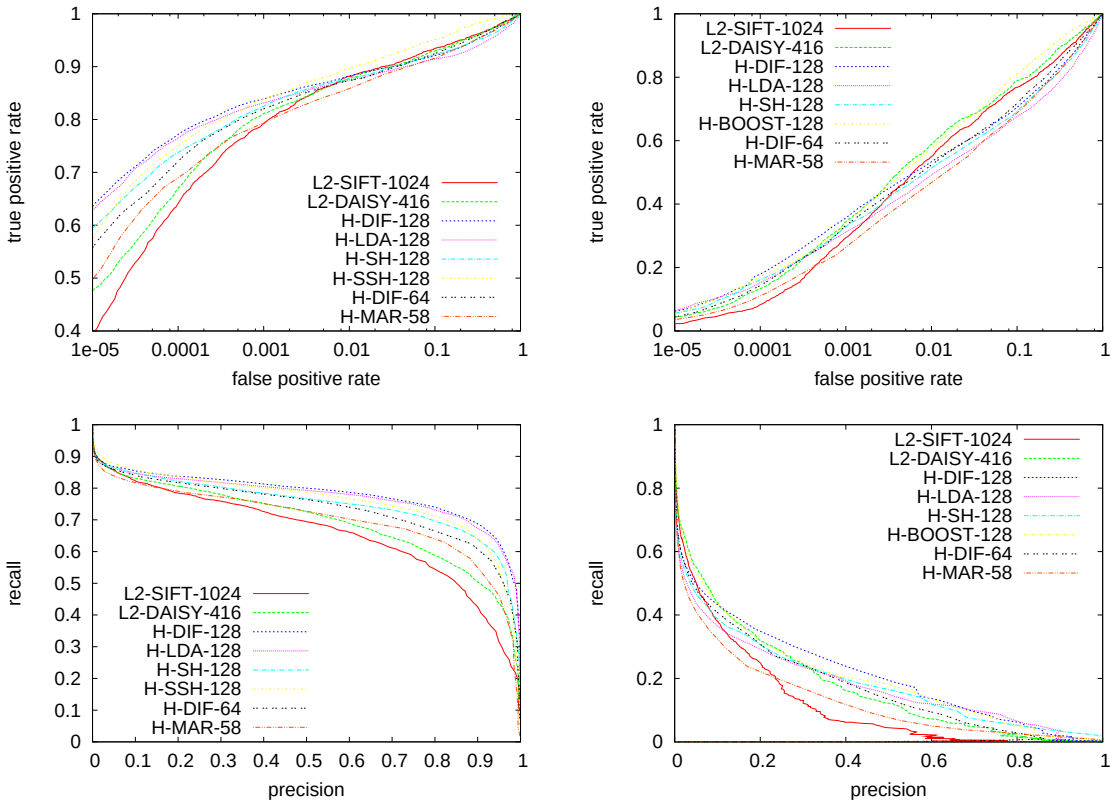
[3] We used this value, since we are primary interested in high precision matches which are needed for calibration purposes. We checked also different values and obtained very similar results.



a-b fountain-P11 with 5923 matches

a-c fountain-P11 with 3053 matches

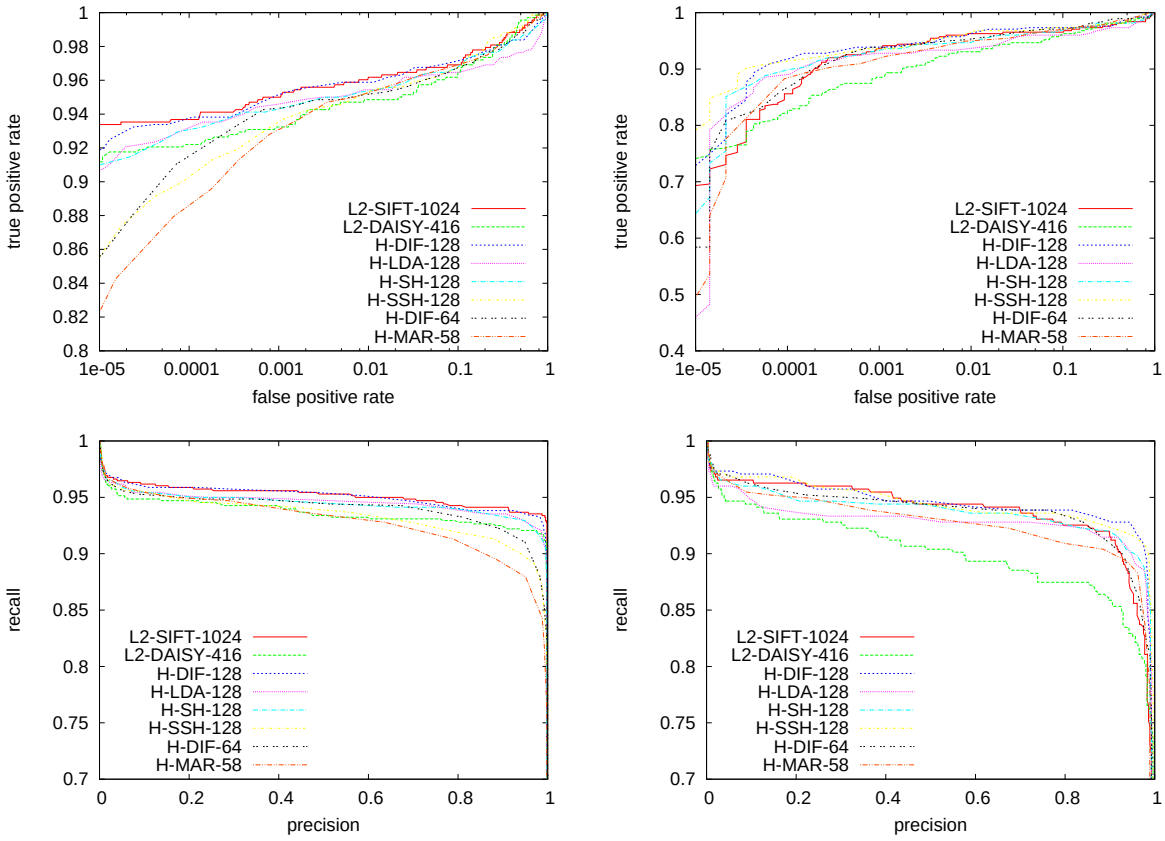
Figure 9: ROC curves (top) and precision v.s. recall (bottom) for binarized and original SIFT as well as DAISY, SH and SSH on the fountain image pairs shown in Figure. 8. When using 128 bit descriptors we systematically outperform all other methods and perform at least similarly when using 64 bit descriptors.



d-e Herz-Jesu-P8 with 3638 matches

d-f Herz-Jesu-P8 with 1546 matches

Figure 10: ROC curves (top) and precision v.s. recall (bottom) for binarized and original SIFT as well as DAISY, SH and SSH on the Herz-Jesu image pairs shown in Figure. 8. When using 128 bit descriptors we systematically outperform all other methods and perform at least similarly when using 64 bit descriptors.



(a) wall img1-img2 with 680 matches

(b) graffity img1-img2 with 375 matches

Figure 11: ROC curves (top) and precision v.s. recall (bottom) for binarized and original SIFT as well as DAISY, SH and SSH. on the image pairs of wall (a) and graffiti (b) from [2].

to SIFT, which was also reported by their authors in [6, 15]. When compared to current state of the art hashing approaches [13] spectral hashing (SH) and similarity-sensitive hashing (SSH), using the same descriptor size (128 bits) we can appreciate a performance boost over the full precision/ FP range. Our DIF projections are slightly better than LDA projections and perform still very well with only 64 bits. On the Mikolajczyk datasets 11 the results do not show a clear direction. This is grounded in the small number of ground truth matches (680 and 375) that make matching confusions less likely and on the fact that the image pairs are relatively easy.

5.2 Large Scale Ground Truth Evaluation

To test our hashing scheme for large scale keypoint retrieval on substantially different images, we calibrated four other datasets depicted in Figures 12, 13, 14 and 15 using SIFT L_2-norm matching ans as described in Section 4.1. The first dataset consists of 71 aerial images (41M pixels), and the other three of 192, 107 and 310 urban images. All datasets contain millions of matching examples and especially the Venice dataset with about 13 million data points covers also interesting situations with strong light and scale changes. The ROC curves are shown in Figs. 12, 13, 14 and 15. Overall, we get an improvement in performance for these large scale datasets, which indicates that our learning scheme generalizes properly ans scales well.

The first three datasets are relatively easy. Baselines in these datasets are small and many of the images are taken under similar light conditions, which is especially true for the aerial dataset in Fig. 12. As a result the improvement of our metric learning is less outspoken than in the last example of Venice (Fig. 15). This dataset contains images from photo community collections take by many different users at different times. One can notice here a significant improvement for 128 bit LDA and DIF projections as well as for 64 bit DIF projections for low false positive rates. More particular, as can be conducted from the graphs, we retrieve the correct keypoint in 83% (78%) of the cases with 128 (64) bits at FP = 0.001 (corre-

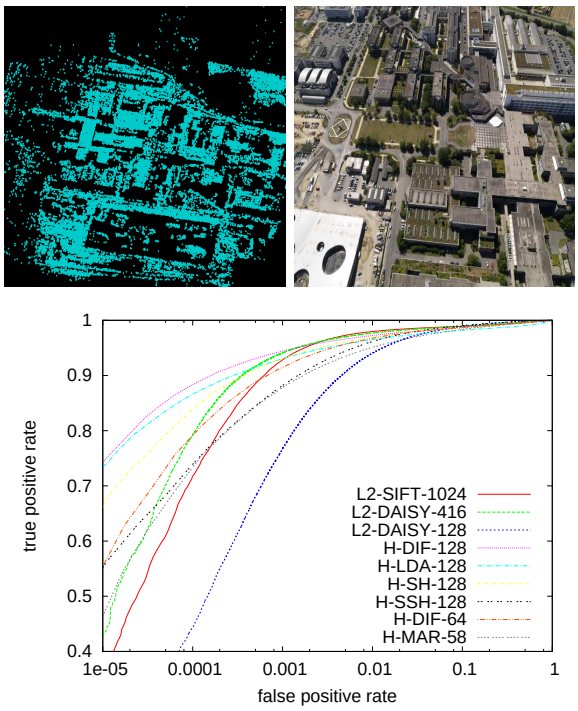


Figure 12: ROC curves for our learned binary descriptors together with original SIFT, DAISY [6], spectral hashing [13] and boosted learning by [10] on an aerial image set with 6,375,139 positive and negative matching examples. Note, that this test image set (show at the top) is very different from our terrestrial image training set also in that more vegetation is present. The good performance H-DIF-16 and H-LDA-16 indicates a good generalization of our learning procedure.

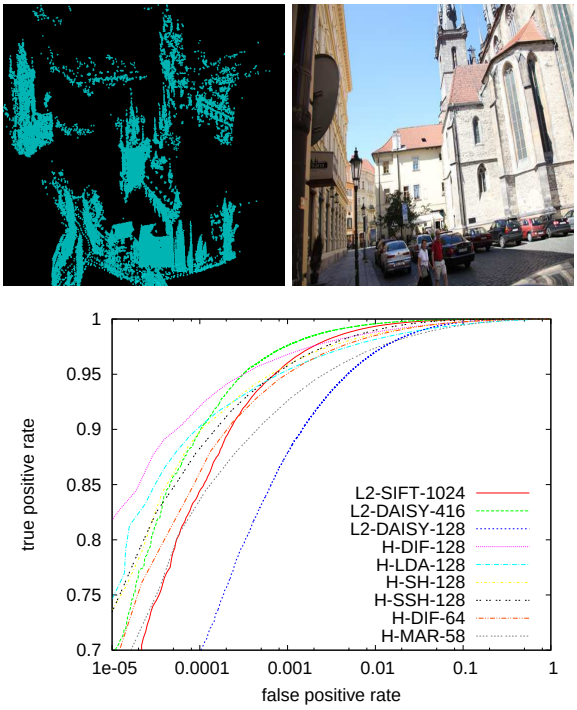


Figure 13: ROC curves for our binary descriptors together with original SIFT, DAISY [6], spectral hashing [13] and boosted learning by [10] on the dataset of Prague with 2,027,389 positive and negative matching examples.

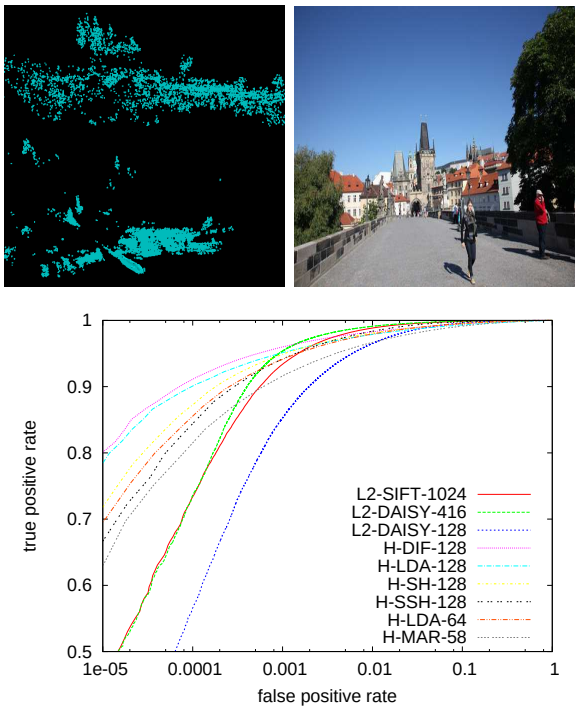


Figure 14: ROC curves for our learned binary descriptors together with original SIFT, DAISY [6], spectral hashing [13] and boosted learning by [10] on the urban dataset of Karl's bridge in Prague with 4,732,375 positive and negative matching examples. This dataset contains a lot of vegetation, which was largely under represented in our training dataset from Lausanne (Fig. 2).

sponding to 12796 false positives in total), which is substantially better that SIFT and DAISY-416 with 56% and 69%, respectively. At the same time we need only 12.5% (6.25%) of the space and bandwidth to store and transfer the descriptors for processing. The difference is much more outspoken if we go to more realistic, lower values of the false positive rate.

If we compare the performance of the descriptors with 128 bits and less, we outperform the other approaches SSH, SH and DAISY-128 over the full false positive range.

The improvement of our metric learning scheme

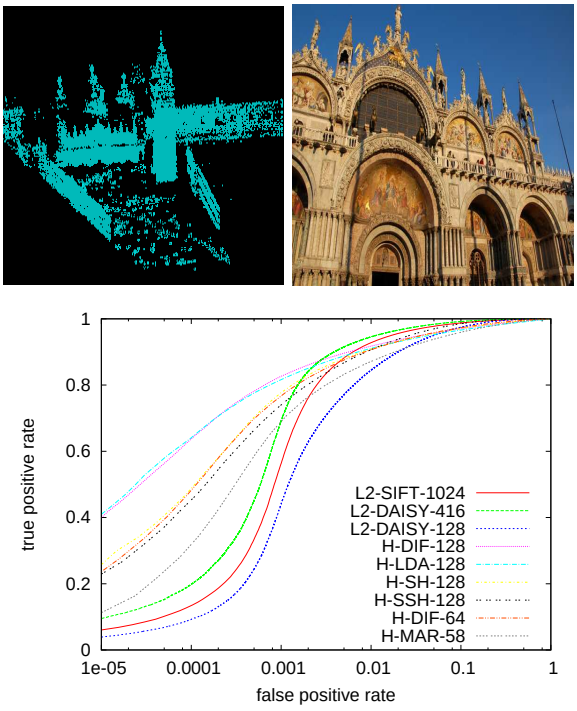


Figure 15: ROC curves for our learned binary descriptors together with original SIFT, DAISY [6, 15], spectral hashing [13] and boosted learning by [10] on the flickr dataset of Venice with 12,796,971 positive and negative matching examples. This dataset contains images taken by different cameras and with different light, weather and seasonal conditions. For this reason and for its size it is the most challenging dataset.

can be explained by the large amount of conjunctive closure matches in our training set(see figures 16 and 17 for example). They are true matches, in that they correspond to the projection of the same physical 3-D point, but may be relatively far apart when compared by SIFT L_2 norm. Our hashing scheme accounts for that and brings those keypoints closer in the Hamming space. At the same time keypoints from different 3-D points are pushed apart, as can be seen in Figure 18. This results in an even greater performance boost over SIFT when wide-baseline and small-baseline is compared as seen in Figures 9 and 10 and when the images contain strong appearance changes as in the Venice dataset shown in Figure 15.

Our evaluation confirms earlier results on the performance of the (52-dimensional) DAISY descriptor [6, 15] when compared to SIFT, which is visible especially in the large scale datasets. To build the DAISY descriptor an extensive optimization of the filter locations, that are used to fill up the descriptor bins, has been performed. This was not done here. Surprisingly, the good low false positve performance of our descriptors when compared to DAISY-416 is consistent and could be explained by the difference in generation the training data (as discussed in Section 4.1) and by the fact that DAISY does not use supervision for its last, quantization step. We think that this is important and show here, as seen in Figure 5, that it leads to a larger performance boost than the un-supervised quantization strategy used by DAISY.

Our experiments show that DIF projections perform slighly better that LDA projections. Both are better that the projections computed by MAR.

5.3 Dependence on Keypoint Detector

Local keypoint descriptors are often highly coupled to keypoint detectores, since computation time can be saved by this strategy. For all evaluation so far we used the SIFT related keypoint detector with is based on Difference of Gaussians (DoG) [1]. DAISY [6] and SURF [3] use other keypoint detectors, which are based on Laplacians and Hessians, respectively. An evaluation on the matching performance for SIFT,

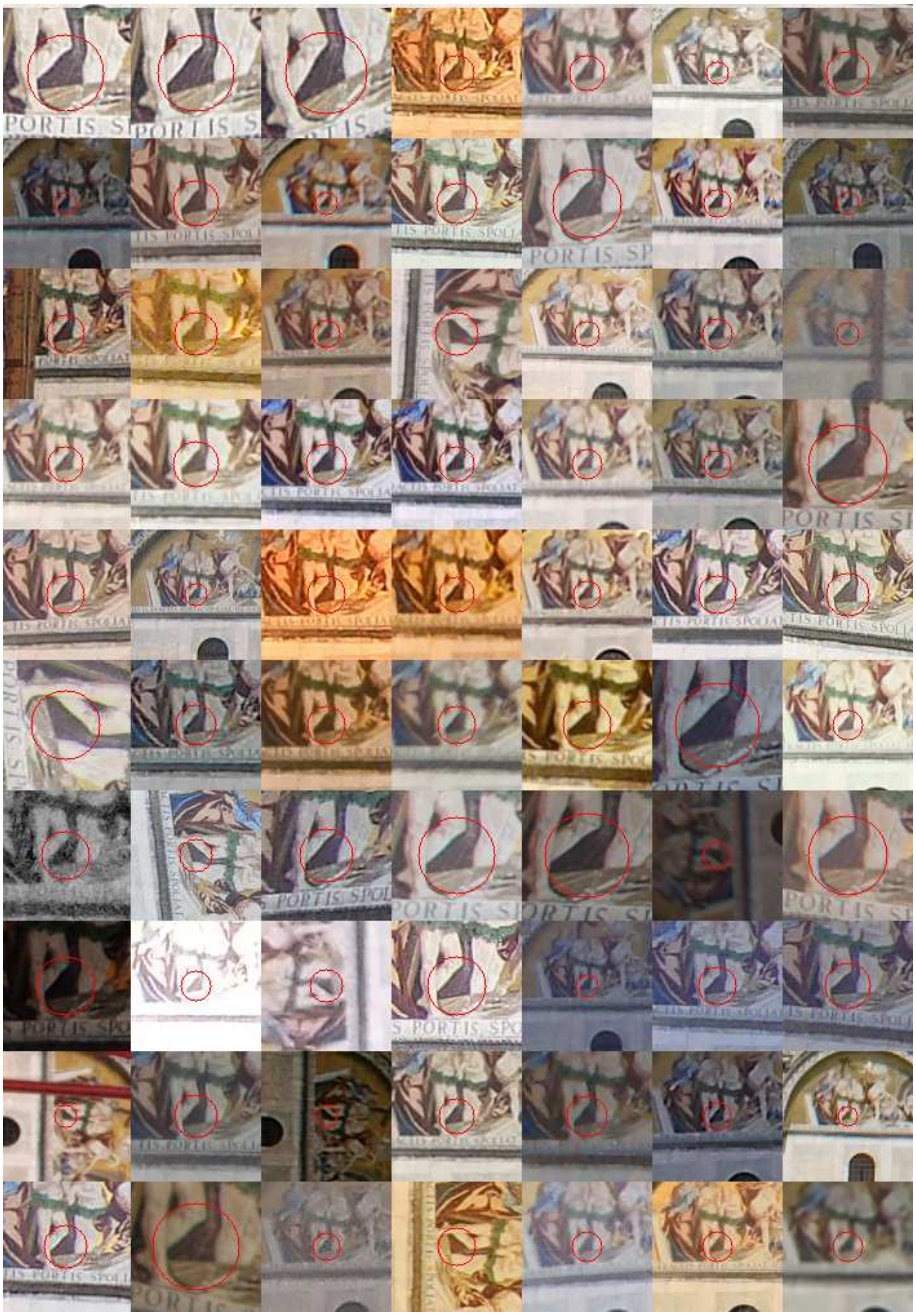


Figure 16: Sample of image patches around the keypoints from a single track, corresponding to the same 3-D point for the Venice dataset in Figure 15. In each patch we show the detected keypoint (DoG) position and denote its scale by a red circle. Note, that the center of the circle is the keypoint location which may be slightly different from the projection of the 3-D point.

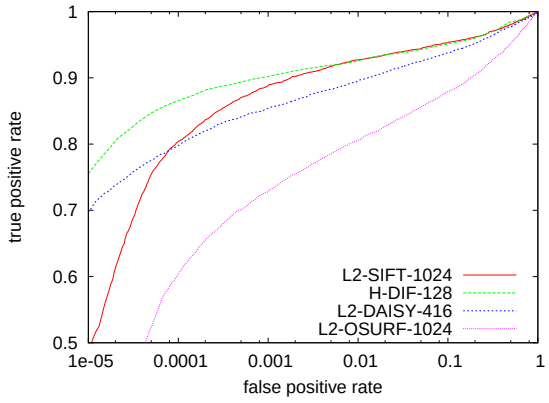


Figure 19: ROC curves for the performance of the descriptors on their own keypoint detector with L2-SIFT-1024 and H-DIF-128 using DoG keypoints, L2-DAISY-416 using Laplacian Keypoints [6] and L2-OSURF [3] using Hessian keypoints. We use 5000 ground truth keypoints on the fountain dataset depicted in Fig. 8 (a-b).

DAISY and SURF *with their own* keypoint detectors is shown in Figure 19. For a fair comparison we sampled for each keypoint detector a constant number of 5000 matches for the fountain (a-b) dataset in Fig. 8. The results show that the DoG keypoint detector performs best and that DAISY gives better results on those keypoints when compared to its own keypoint detections.

6 Conclusions

We presented a novel and simple approach to produce a binary string from a SIFT descriptor. Our approach first aligns the SIFT descriptors according to the problem specific covariance structure. In the resulting vector space, all SIFT descriptors have diagonal covariance. We can then estimate reliable thresholds that perform the binarization according to an appropriate cost function. This approach is very fast and can be used for many other applications for which similar training data is available.

We showed in this paper that this very simple and general approach leads to outstanding matching results with a very compact descriptor. Our resulting binary descriptor performs better than original SIFT [1, 31] and DAISY [6, 15] in the low false positive range, which is the interesting range for large scale keypoint retrieval applications. Thereby our 128 bit version requires only $\approx 10\%$ of the size SIFT uses to ($\approx 25\%$ of the DAISY size, respectively) to describe keypoints. When compared to locality-sensitive hashing [10] and spectral hashing proposed by Weiss *et.al.* [13], which use the same number of bits to encode keypoints, our descriptors perform better in the whole false positive range. This is also true if we compare to a reduced size DAISY with 128 bits.

Very good performance for low false positive rates can be obtained by using as few as 64 bits (H-DIF-64), which makes is possible to search efficiently in a large database. Matching is very fast for binary descriptors even for exhaustive search, since only a XOR followed by a bit count is needed to compute the Hamming distance (in some modern CPUs, bit counting is implemented as a single instruction). Moreover, binary descriptors with the Hamming metric can be indexed efficiently on existing database management systems, a direction we intend to explore in future research. We believe that matching of our binary representations can be performed very fast even on mobile devices, and release our binarizations for SIFT into the public domain [37].

Philosophically, our approach addresses the gap between *modeling* and *learning* in feature descriptor design. The recent trend in computer vision literature has been to construct feature descriptors that would theoretically be invariant to certain transformations such as rotations or affine transformations. However, such transformations are only approximations of the real image formation model, and thus the descriptor is never truly invariant. Augmenting it with a metric learning approach, it is possible to learn invariance to typical transformations that may appear in a natural scene. It would be interesting to explore the tradeoff between how much effort should be invested in modeling invariance versus learning it from examples.

Interesting further research could look at other descriptors such as DAISY [6], SURF [3] or BRIEF [38],

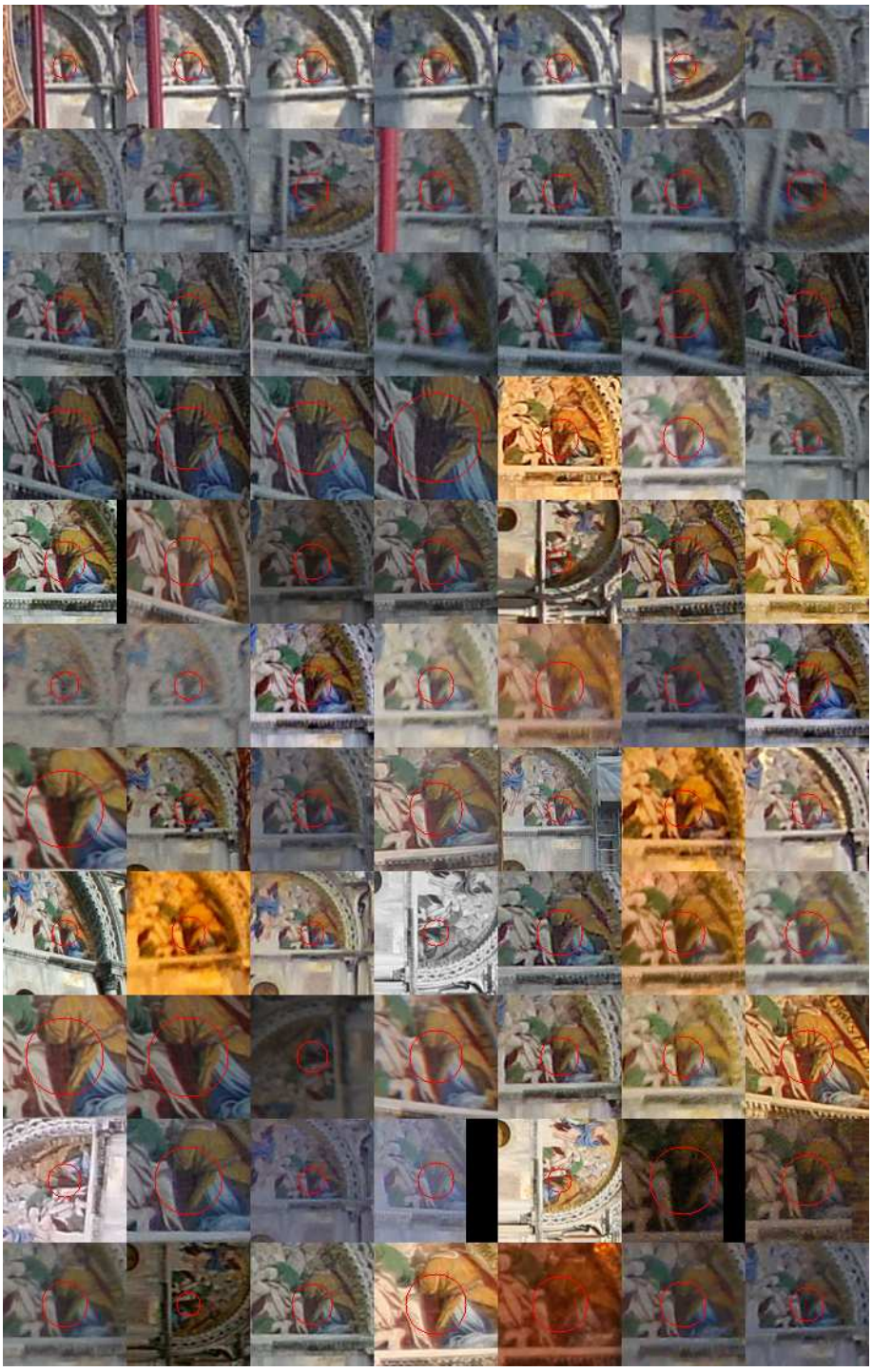


Figure 17: More keypoints as in Fig. 16.

which are faster to compute and to learn a similar binarization. We also plan to investigate the performance of an additional network layer to reduce the size of our current binary descriptors even further and without loss in performance.

Acknowledgments

We would like to thank Matthew Brown for useful discussion, Simon Winder for providing his DAISY binary and for testing DAISY on the fountain sequence and Andrea Vedaldi for his SIFT implementation. The authors further acknowledge Flickr users for providing their images for the Venice and for parts of the Prague and Lausanne datasets as well as SenseFly (www.sensefly.com) for capturing the aerial image of the EPFL in Lausanne.

References

[1] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *IJCV*, vol. 20, no. 2, pp. 91–110, 2004.

[2] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *IJCV*, vol. 65, no. 1/2, pp. 43–72, 2005.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," *Computer Vision and Image Understanding*, vol. 10, no. 3, pp. 346–359, 2008.

[4] E. Tola, V. Lepetit, and P. Fua, "Daisy: an Efficient Dense Descriptor Applied to Wide Baseline Stereo," *Trans. PAMI*, vol. 32, no. 5, pp. 815–830, 2010.

[5] T. Tuytelaars and C. Schmid, "Vector quantizing feature space with a regular lattice," *Proc. ICCV*, 2007.

[6] S. Winder, G. Hua, and M. Brown, "Picking the best DAISY," in *Proc. CVPR*, June 2009.

[7] K. Mikolajczyk and C. Schmid, "A Performance Evaluation of Local Descriptors," *Trans. PAMI*, vol. 27, no. 10, pp. 1615–1630, 2004.

[8] G. Hua, M. Brown, and S. Winder, "Discriminant embedding for local image descriptors," in *Proc. ICCV*, 2007.

[9] A. Gionis, P. Indik, and R. Motwani, "Similarity Search in High Dimensions via Hashing," in *International Conference on Very Large Databases*, 2004.

[10] G. Shakhnarovich, "Learning Task-Specific Similarity," Ph.D. dissertation, MIT, 2005.

[11] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Proc. NIPS*, 2009, pp. 1042–1050.

[12] M. Raginsky and S. Lazebnik, "Locality-Sensitive Binary Codes from Shift-Invariant Kernels," *Advances in neural information processing systems*, 2009.

[13] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in neural information processing systems*, vol. 21, pp. 1753–1760, 2009.

[14] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Boosting algorithms as gradient descent," in *Proc. NIPS*. MIT Press, 2000, pp. 512–518.

[15] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, no. PrePrints, 2010.

[16] S. Winder and M. Brown, "Learning Local Image Descriptors," in *Proc. CVPR*, Minneapolis, MI, June 2007.

[17] V. Chandrasekhar, G. Takacs, D. M. Chen, S. S. Tsai, R. Grzeszczuk, and B. Girod, "Chog: Compressed histogram of gradients a low bit-rate feature descriptor," in *CVPR*, 2009, pp. 2504–2511.

[18] K. Mikolajczyk and C. Schmid, "A Performance Evaluation of Local Descriptors," in *Proc. CVPR*, June 2003, pp. 257–263.

[19] K. Mikolajczyk and J. Matas, "Improving descriptors for fast tree matching by optimal linear projection," in *Proc. ICCV*, 2007.

[20] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: a large dataset for non-parametric object and scene recognition," *Trans. PAMI*, vol. 30, no. 11, pp. 1958–1970, 2008.

[21] H. Jégou, M. Douze, and C. Schmid, "Packing Bag-of-Features," in *Proc. ICCV*, 2009.

[22] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Video genome," Tech. Rep. arXiv:1003.5320v1, 2010.

[23] A. Bronstein, M. Bronstein, M. Ovsjanikov, and L. Guibas, "Shape Google: geometric words and expressions for invariant shape retrieval," *ACM TOG*, 2010.

[24] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. ECCV*, 2008, pp. 304–317.

[25] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *Trans. PAMI*, 2010.

[26] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *Proc. CVPR*, 2010.

[27] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *European Conference on Computational Learning Theory*, 1995, pp. 23–37.

[28] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[29] A. Demiriz, K. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *Machine Learning*, vol. 46, no. 1, pp. 225–254, 2002.

[30] C. Strecha, T. Pylvanainen, and P. Fua, "Dynamic and Scalable Large Scale Image Reconstruction," in *Proc. CVPR*, San Francisco, CA, June 2010.

[31] A. Vedaldi, "An open implementation of the SIFT detector and descriptor," UCLA CSD, Tech. Rep. 070012, 2007.

[32] K. Heath, N. Gelfand, M. Ovsjanikov, M. Aanjaneya, and L. J. Guibas, "Image Webs: Computing and Exploiting Connectivity in Image Collections," in *Proc. CVPR*, 2010.

[33] A. Torralba, R. Fergus, and Y. Weiss, "Small Codes and Large Databases for Recognition," in *Proc. CVPR*, June 2008.

[34] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen, "On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery," in *Proc. CVPR*, Anchorage, AK, 2008.

[35] C. Strecha, "Multi-view evaluation - http://cvlab.epfl.ch/data," 2008.

[36] C. Strecha and P. Fua, "Local keypoint evaluation - http://cvlab.epfl.ch/data/," 2010.

[37] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua, "Ldahash - http://cvlab.epfl.ch/software," 2010.

[38] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF binary robust independent elementary features," in *ECCV to appear*, 2010.

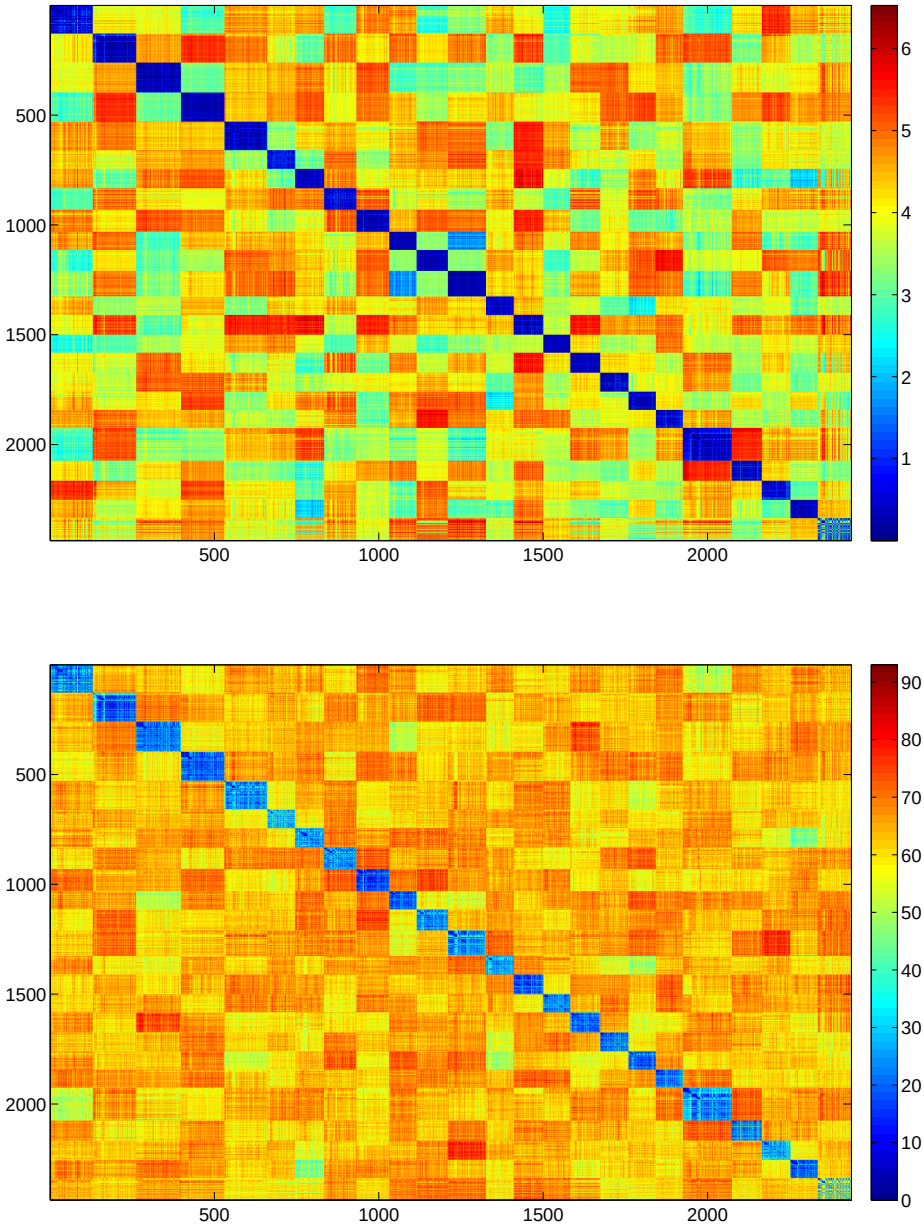


Figure 18: Confusion matrix for the 2437 keypoints from the largest 24 tracks (3-D points) in the dataset of Fig. 15. Two of these tracks are depicted by Figs. 16 and 17. One can see the block diagonal structure, where each block corresponds to one tracks. At the top one can see the SIFT L_2 norm color coded between minimum and maximum. At the bottom one can see the same for the H-DIF-128 binarization, which exhibits far less confusion.