Fast Quantization and Matching of Histogram-Based Image Features

Yuriy A. Reznik[a], Vijay Chandrasekhar[b], Gabriel Takacs[b], David M. Chen[b],
Sam S. Tsai[b], and Bernd Girod[b]

[a]Qualcomm Inc., 5775 Morehouse Dr., San Diego, CA 92121, USA;
[b]Information Systems Laboratory, Stanford University, Stanford, CA 94305, USA

ABSTRACT

We review construction of a Compressed Histogram of Gradients (CHoG) image feature descriptor, and study quantization problem that arises in its design. We explain our choice of algorithms for solving it, addressing both complexity and performance aspects. We also study design of algorithms for decoding and matching of compressed descriptors, and offer several techniques for speeding up these operations.

Keywords: Mobile Visual Search, Feature Descriptors, Quantization, SIFT, SURF, CHoG .

1. INTRODUCTION

Mobile phones have evolved into powerful image and video processing devices, equipped with high-resolution cameras, color displays, and hardware-accelerated graphics. They are also equipped with location sensors, GPS receivers, and connected to broadband wireless networks allowing fast transmission of information. This enables a class of applications which use the camera phone to initiate search queries about objects in visual proximity to the user. Such applications can be used for identifying products, comparison shopping, finding information about movies, CDs, real estate or products of the visual arts. *Google Goggles*,[1] *Point and Find*,[2] and *Snaptell*[3] are examples of recently developed commercial applications. For these applications, a query photo is taken by a mobile device and compared against previously stored database photos. A set of image feature descriptors is used to assess the similarity between the query photo and each database photo. This feature set needs to be robust against geometric and photometric distortions encountered when the user takes the query photo from an arbitrary viewpoint in an unknown lighting environment.

The size of the data sent over the wireless network needs to be as small as possible to reduce latency and improve user experience. One approach to the problem is to transmit a JPEG compressed query image over the network, but this might be prohibitively expensive at low uplink speeds. An alternate approach is to extract feature descriptors on the phone, compress the descriptors and transmit them over the network as illustrated in Figure 1. Such an approach has been demonstrated to reduce the amount of transmission data significantly.[4, 5] Yet another approach is to perform descriptor extraction and search in a cache of the database stored on a mobile phone. Such architecture reduces frequency of requests to a remote database, and it was shown to be most promising for the design of mobile augmented reality applications.[6]

1.1 Related Prior Work

Scale Invariant Feature Transform (SIFT),[7] Speeded Up Robust Features (SURF),[8] Gradient Location and Orientation Histogram (GLOH),[9] and Compressed Histograms of Gradients (CHoG)[4] are popular feature descriptors proposed in the literature. The review paper by Mikolajczyk *et al.*[9] compares the performance of several descriptors. This review, however, does not take the bit-rate of descriptors into account.

Low bit-rate feature descriptors are of increasing interest to the computer vision community. Often, feature vectors are reduced by decreasing the dimensionality of descriptors via Principle Component Analysis (PCA) or

Further author information: (Send correspondence to B Girod.)
Y.A.Reznik: E-mail: yreznik@qualcomm.com
V.Chandrasekhar, G.Takacs, D.Chen, S.Tsai, B.Girod: E-mail: {vijayc,gtakacs,dmchen,sststai,bgirod}@stanford.edu

Applications of Digital Image Processing XXXIII, edited by Andrew G. Tescher,
Proc. of SPIE Vol. 7798, 77980L · © 2010 SPIE · CCC code: 0277-786X/10/$18
doi: 10.1117/12.862362

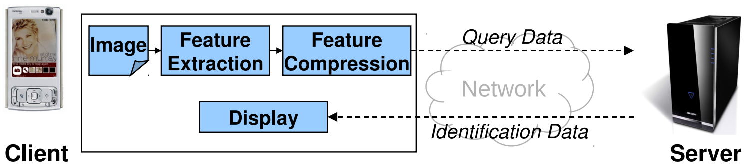


Figure 1. A mobile CD cover recognition system. The server is located at a remote location. Feature descriptors are extracted on the mobile-phone and query feature data is sent over the network. Once the CD cover is recognized on the server, identification data is sent back to the mobile-phone.

Linear Discriminant Analysis (LDA).[10–12] V. Chandrasekhar *et al.*[13] study dimensionality reduction and entropy coding of SIFT and SURF descriptors. Yeo *et al.*[15] and Shakhnarovich *et al.*[16] reduce the bit-rate of descriptors by using projections on SIFT descriptors to build binary hashes. As part of the MPEG-7 standard, Brasnett and Bober[17] propose a 60-bit feature descriptor, which will also use as a reference.

In our prior work,[4,5] we propose a framework for computing low bit-rate feature descriptors called CHoG. Gradient histograms are subsequently quantized and compressed using fixed and variable length codes. Original design of CHoG descriptors[4] used Huffman trees for quantization of histograms. In,[5] we introduced type-coding and generalized Lloyd VQ techniques resulting in more efficient solutions of histogram quantization problem. This paper is an extension of study,[5] offering more detailed description of the type-coding scheme, analysis of its performance, and focusing on the design of fast and memory-efficient algorithms for histogram quantization and matching.

1.2 Outline

This paper is organized as follows. In Section 2 we give an overview of the design of a compressed histogram of gradients (CHoG) descriptor. In Section 3 we discuss quantization problem that arises in this design and explain our choices of algorithms for solving it. In Section 4, we discuss design of fast algorithms for decoding and matching of our descriptors. In Section 5 we analyze performance of our quantization scheme and feature descriptor. Conclusions are drawn in Section 6.

2. COMPRESSED HISTOGRAM OF GRADIENTS DESCRIPTOR

In this section we review main steps in the design of a Compressed Histogram of Gradients (CHoG) descriptor.[4,5]

2.1 Patch Extraction and Spatial Binning

We start with a canonical patch, extracted around an interest point at the detected scale and orientation. As suggested by Mikolajczyk and Schmid,[9] we model illumination changes to the patch appearance by a simple affine transformation $aI + b$ of the pixel intensities, which is compensated by normalizing the mean and standard deviation of the pixel values of each patch. Local image gradients d_x and d_y are computed using a centered derivative filter kernel $[-1, 0, 1]$.

The patch is then divided into several localized spatial cells. In our design, we use a modification of DAISY configurations proposed in[12,30]. In the original design[12], the patch is divided in several disjoint localized cells. In this work, we use overlapping (”soft”) spatial bins, as shown in Figure 2. The soft assignment is made such that in computing histograms each gradient contributes to multiple spatial bins. We use normalized Gaussian weights that sum to 1 at each location. A value of σ for the Gaussian that works well is $d_{min}/3$, where d_{min} is the minimum distance between bin centers in the DAISY configuration. Spatial binning improves the performance of the descriptor by making it more robust to the interest point localization error.[7]

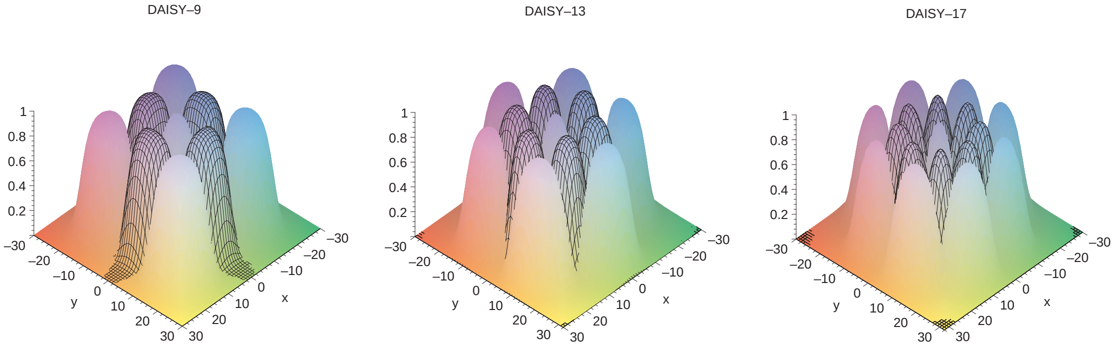
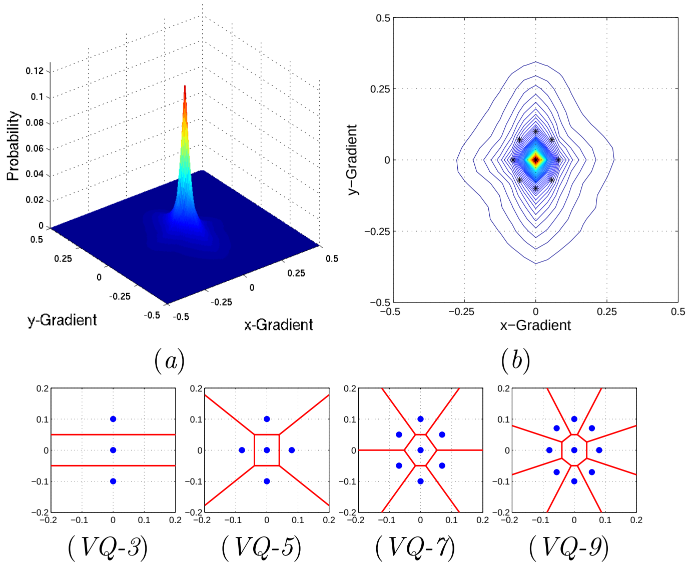


Figure 2. DAISY configurations with $K = 9, 13, 17$ spatial cells. We use Gaussian-shaped overlapping (soft) binning. For picture clarity, weighting functions of inner and outer cells are shown without grid covering.



(a) (b)

$(VQ\text{-}3)$ $(VQ\text{-}5)$ $(VQ\text{-}7)$ $(VQ\text{-}9)$

Figure 3. The joint (d_x, d_y) gradient distribution (a) over a large number of cells, and (b), its contour plot. The greater variance in y-axis results from aligning the patches along the most dominant gradient after interest point detection. The quantization bin constellations VQ-3, VQ-5, VQ-7 and VQ-9 and their associated veronoi cells are shown at the bottom.

2.2 Quantization of Gradients

Our next task is to quantize histograms of gradients in each spatial cell. Let $P_{D_x,D_y}(d_x, d_y)$ be the normalized joint (x, y)-gradient histogram in each spatial bin. We propose coarsely quantizing the 2D gradient histogram and capturing the histogram directly into the descriptor. We approximate $P_{D_x,D_y}(d_x, d_y)$ as $\hat{P}_{\hat{D}_x,\hat{D}_y}(\hat{d}_x, \hat{d}_y)$ for $(\hat{d}_x, \hat{d}_y) \in S$, where S represents a small number of quantization centroids or bins as shown in Figure 3.

We use histogram binning schemes that exploit the underlying gradient statistics observed in patches extracted around interest points, as shown in Figure 3. We perform a Vector Quantization (VQ) of the gradient distribution to a small set of bin centers, S, shown in Figure 3. We call these bin configurations VQ-3, VQ-5, VQ-7 and VQ-9. Similar to soft spatial binning, we assign each (d_x, d_y) pair to multiple bin centers with normalized Gaussian weights. We use $\sigma = q_{min}/3$, where q_{min} is the minimum distance between centroids in the VQ bin configurations shown in Figure 3. As we increase the number of bin centers, we obtain a more accurate approximation of the gradient distribution and the performance of the descriptor improves.[4] All soft binning weights are pre-computed which enables fast computation of the descriptor.

2.3 Second-stage Quantization and Encoding

Our final task is to quantize and encode histograms from spacial cells. The goal is to represent these histograms using small number of bits while maintaining high precision of their representation. We perform independent quantization of histograms in each cell. The resulting codebook indices are then encoded using fixed-length or arithmetic codes. The final bitstream of the feature descriptor is formed as a concatenation of codes representative of histograms in each cell.

The quantization process that we employ at the this stage is explained in greater detail in the next section. It is one of the most computationally intensive steps in the descriptor extraction process, requiring us to pay attention to complexity in our choice of algorithms.

3. QUANTIZATION OF HISTOGRAMS

We start by offering a simple mathematical description of the problem that we encounter in the final stage of design of CHoG descriptors.

3.1 Description of the Problem

Let m represent the number of histogram bins. By Ω_m we denote a set of all possible m-ary probability distributions:

$$\Omega_m = \left\{ [\omega_1, \ldots, \omega_m] \in \mathbb{R}^m \middle| \forall i : \omega_i \geqslant 0 \, , \textstyle\sum_i \omega_i = 1 \right\} . \tag{1}$$

This produces a compact subset of $\mathbb{R}^{m-1}$, commonly known as the unit $(m-1)$-*simplex*.[19]

Let $p \in \Omega_m$ be an input distribution, and let $Q \subset \Omega_m$ be a set of distributions that we will be able to reproduce. We will call elements of Q *reconstruction points* or *centers* in Ω_m. We assume that $|Q| < \infty$ and that its elements can be enumerated and encoded. When using fixed-rate encoding the rate becomes $R(Q) = \lceil \log_2 |Q| \rceil$ bits. By $d\,(p,q)$ we denote a *distance measure* between distributions $p, q \in \Omega_m$. For example, it can be an r-th order L-norm:

$$d_r(p,q) = ||p-q||_r = \left(\sum_i |p_i - q_i|^r \right)^{1/r} , \tag{2}$$

Kullback-Leibler (KL) distance:

$$d_{\mathrm{KL}}(p,q) = D(p||q) = \sum_i p_i \log_2 \frac{p_i}{q_i} , \tag{3}$$

or symmetric KL-distance (also known as Jeffrey's divergence):

$$d_{\mathrm{J}}(p,q) = D(p||q) + D(q||p) . \tag{4}$$

In the context of our application, symmetric KL-distance will be of most interest.[4]

In a simplest form, the problem of quantization of distribution $p \in \Omega_m$ can be understood as task of finding the set $Q : |Q| \leqslant 2^R$, such that the *maximal distance* to the nearest reconstruction point is minimal:

$$d^*(\Omega_m, R) = \inf_{\substack{Q \subset \Omega_m \\ |Q| \leqslant 2^R}} \max_{p \in \Omega_m} \min_{q \in Q} d(p,q) . \tag{5}$$

When it is further assumed that distributions $p \in \Omega_m$ are produced by some random process with density θ over Ω_m, the quantization problem can be formulated as minimization of the *expected distance* to the nearest reconstruction point:

$$\bar{d}(\Omega_m, \theta, R) = \inf_{\substack{Q \subset \Omega_m \\ |Q| \leqslant 2^R}} \mathbf{E}_{\substack{p \in \Omega_m \\ p \sim \theta}} \min_{q \in Q} d(p,q) , \tag{6}$$

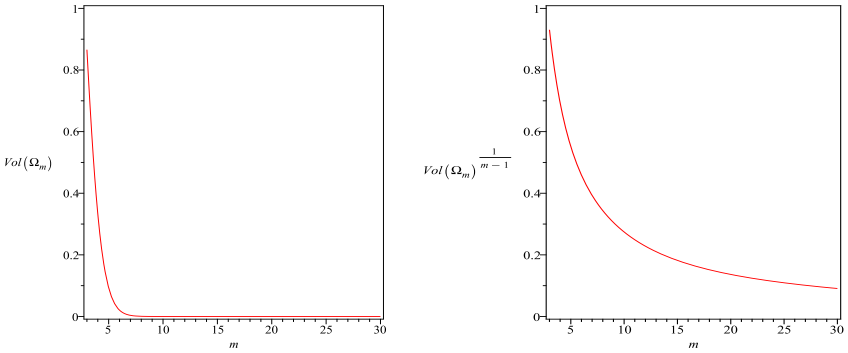


Figure 4. Volume of the probability simplex Ω_m (left) and the associated leading factor in asymptotic expression for quantization error $d^*(\Omega_m, R)$ (right). Both quantities become smaller as the number of dimensions m increases.

Both settings are standard in quantization theory.[20] The distinctive part of our problem is a particular shape of set Ω_m that we need to quantize. For example, it can be shown (cf.[19]), that the volume of unit $m-1$-simplex Ω_m is rapidly decaying with m:

$$\mathrm{Vol}(\Omega_m) = \frac{a^k}{k!}\sqrt{\frac{k+1}{2^k}}\,\Bigg|_{\substack{k=m-1\\ a=\sqrt{2}}} = \frac{\sqrt{m}}{(m-1)!}\,. \tag{7}$$

This suggests, that in high dimensions, quantization error (5) or (6) achievable in our case should be much smaller compared to one resulting from solving quantization problem for the unit cube $[0,1]^{m-1}$. The precise amount of reduction of quantization error is actually

$$\sqrt[m-1]{\mathrm{Vol}(\Omega_m)} = \sqrt[m-1]{\frac{\sqrt{m}}{(m-1)!}} = \frac{e}{m} + O\left(\frac{1}{m^2}\right) \tag{8}$$

which appears as a factor in achievable (high-rate regime) characteristic of the quantizer [20, Theorem 10.7]:

$$d_r^*(\Omega_m, R) \sim C_{m-1,r}\,\sqrt[m-1]{\mathrm{Vol}(\Omega_m)}\;2^{-\frac{R}{m-1}}\,, \tag{9}$$

where $C_{m-1,r}$ are some known constants, for example, $C_{m-1,\infty} = \frac{1}{2}$ (for any m), $C_{2,1} = \frac{1}{\sqrt{2}}$, $C_{2,2} = \sqrt{\frac{2}{3\sqrt{3}}}$, etc.

In other words, coding of distributions (histograms of gradients of image features) has some interesting properties and advantages from quantization-theory point of view. More details on this subject can be found in.[25]

We next describe design of a practical algorithm for solving histogram quantization problem. In order to be suitable for use in mobile devices (mobile phones, cameras, etc.) such algorithm should require small amount of memory and have low computational complexity.

3.2 Choice of Codebook

The need for small memory footprint forces us to focus on algorithms that do not explicitly store the set of reconstruction points (or *codebook*) Q. Instead, such codebook must have some regular structure, allowing on-the-fly computation of reconstruction points.

Specifically, codebook Q must contain a set of m-ary distributions $q = [q_1, \ldots, q_m]$. In our design, we define values q_i as fractions $q_i = k_i/n$, where n, and k_i $(i = 1, \ldots, m)$ are some integers. This leads to the following set of distributions:

$$Q_n = \left\{ [q_1, \ldots, q_m] \in \mathbb{Q}^m \,\middle|\, \forall i : q_i = \tfrac{k_i}{n}, k_i \geqslant 0, \textstyle\sum_i k_i = n \right\}, \tag{10}$$

The parameter n controls the density and the number of points in Q_n. We show several examples of such sets in $m = 3$ dimensions in Figure 5.

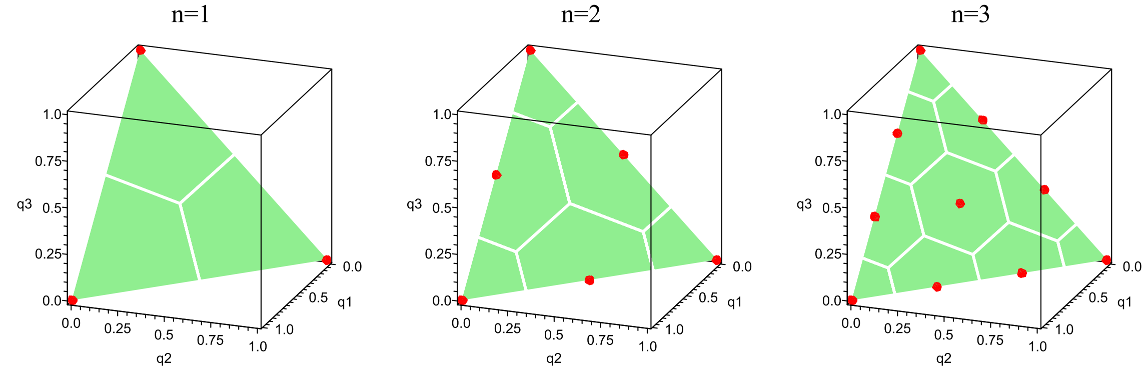


Figure 5. Examples of type lattices and their Voronoi partitions in 3 dimensions ($m = 3, n = 1, 2, 3$). Green triangle shows the body of probability simplex Ω_m that we need to quantize.

We note that points $q \in Q_n$ coincide with the definition of *types* in universal source coding theory.[21] Due to this analogy, we will refer to points $q \in Q_n$ as *types*, and will call the entire set Q_n a *type lattice*.

From the point of view of lattice theory, the set Q_n can be understood as a bounded subset of so-called lattice A_n.[23] It is not the most dense lattice available (e.g. if compared to E_8 or Λ_{24} lattices[23] in dimensions 8 and 24), and as such it is rarely used in practice. However, in the context of our problem, this lattice becomes very convenient: it is remarkably easy to construct, and it can be easily bounded to fill the simplex-shaped subset of $\mathbb{R}^{m-1}$ that we need to quantize.

3.3 Finding Nearest Reconstruction Point

In order to find the nearest type in Q_n we use the following algorithm*:

Algorithm 1. *Given p, n, find parameters $k_1, \ldots, k_m$ of the nearest point $q = \left[\frac{k_1}{n}, \ldots, \frac{k_m}{n}\right]$:*

1. *Compute values ($i = 1, \ldots, m$)*

$$k'_i = \left\lfloor np_i + \tfrac{1}{2} \right\rfloor \, , \quad n' = \textstyle\sum_i k'_i \, .$$

2. *If $n' = n$ the nearest type is given by: $k_i = k'_i$. Otherwise, compute errors*

$$\delta_i = k'_i - np_i \, ,$$

and sort them such that

$$-\tfrac{1}{2} \leqslant \delta_{j_1} \leqslant \delta_{j_2} \leqslant \ldots \leqslant \delta_{j_m} \leqslant \tfrac{1}{2} \, ,$$

3. *Let $\Delta = n' - n$. If $\Delta > 0$ then decrement Δ values k'_i with largest errors*

$$k_{j_i} = \left[\begin{array}{ll} k'_{j_i} & j = i, \ldots, m - \Delta - 1 \, , \\ k'_{j_i} - 1 & i = m - \Delta, \ldots, m \, , \end{array} \right.$$

otherwise, if $\Delta < 0$ increment $|\Delta|$ values k'_i with smallest errors

$$k_{j_i} = \left[\begin{array}{ll} k'_{j_i} + 1 & i = 1, \ldots, |\Delta| \, , \\ k'_{j_i} & i = |\Delta| + 1, \ldots, m \, . \end{array} \right.$$

4. *Return quantities $k_1, \ldots, k_m$.*

*This algorithm is similar in concept to Conway and Sloane's quantizer for lattice A_n.[24] However, our algorithm is much simpler, and produces points that are naturally constrained to the unit simplex. No boundary checks are needed.

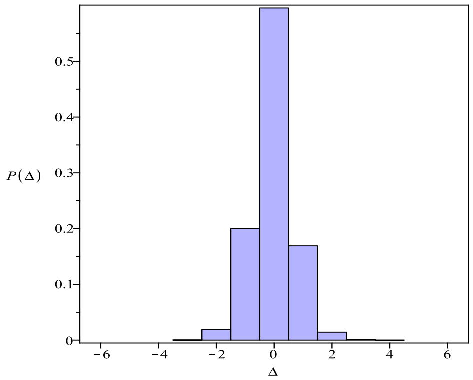


Figure 6. Typical distribution of values Δ in Algorithm 1. Observed with $m = 9$ after computing descriptors for patches in *Liberty* dataset.[12] With larger sets we expect this distribution to be more symmetric.

The most complex step in this algorithm is a sorting operation in Step 2. Using traditional fast sorting procedures, such as quicksort, the result is usually obtained after $O(m \log m)$ comparisons. However, in our situation, we don't really need to perform full sort: in Step 3 we only need to know $|\Delta|$ smallest or largest elements. Finding them can be accomplished by a much simpler procedure, shown below as Algorithm 2.

Algorithm 2. *Given quantities* $\delta_1, \ldots, \delta_m$*, find indices* $j_1, \ldots, j_\Delta$ *of their* Δ *smallest elements:* $\delta_{j_1} \leqslant \ldots \leqslant \delta_{j_\Delta} \leqslant \max_{j \notin \{j_1, \ldots, j_\Delta\}} \delta_j$*:*

```
1. Set $j_i = i$, $i = 1, \ldots, m$.
2. For $k = 1, \ldots, \Delta$ do the following
    (a) find $i$ such that $\delta_{j_i} = \min\left\{\delta_{j_{k+1}}, \ldots, \delta_{j_m}\right\}$;
    (b) swap elements $j_k$ and $j_i$.
3. Return indices $j_1, \ldots, j_\Delta$
```

This algorithm requires $O(m|\Delta|)$ comparisons. This is not the fastest selection algorithm available, but it works well when parameter Δ is small. This is precisely the case in our situation. We show typical distribution of quantities Δ in Figure 6.

3.4 Enumeration and Encoding

The number of types in lattice Q_n depends on the parameter n. It is essentially the number of partitions of n into m terms $k_1 + \ldots + k_m = n$:

$$|Q_n| = \binom{n+m-1}{m-1}. \tag{11}$$

In order to encode a type with parameters $k_1, \ldots, k_m$, we first need to obtain its unique index $\xi(k_1, \ldots, k_m)$. We propose to compute it as follows:

$$\xi(k_1, \ldots, k_m) = \sum_{j=1}^{m-2} \sum_{i=0}^{k_j - 1} \binom{n - i - \sum_{\ell=1}^{j-1} k_\ell + m - j - 1}{m - j - 1} + k_{n-1}\,. \tag{12}$$

This formula follows by induction (starting with $m = 2, 3$, etc.), and it implements lexicographic enumeration of types. For example:

$$\begin{aligned} \xi(0, 0, \ldots, 0, n) &= 0\,, \\ \xi(0, 0, \ldots, 1, n-1) &= 1\,, \\ &\ldots \\ \xi(n, 0, \ldots, 0, 0) &= \tbinom{n+m-1}{m-1} - 1\,. \end{aligned}$$

With precomputed array of binomial coefficients, the computation of an index by using this formula requires $O(n)$ operations. The amount of memory needed to store such an array of coefficients is $O(n\,m)$ words. This is much smaller compared to $O(m\,|Q_n|) = O\left(n^m\right)$ words needed to store the entire codebook. We note, that in the context of our application, we typically use lattices with $n = O(m)$.

Once index is computed, it is transmitted either by using fixed rate codes (direct binary representation of an index) or by using arithmetic codes. In both cases, the achievable rate satisfies:

$$R(n) \leqslant \left\lceil \log_2 \tbinom{n+m-1}{m-1} \right\rceil . \tag{13}$$

3.5 Adding Bias

In our final implementation of type quantizer we have found it useful to introduce slight shift of reconstruction points towards the middle of the simplex. This is done with the help of the *bias parameter* $\beta > 0$:

$$q_i' = \frac{k_i + \beta}{n + \beta m}\,, \quad i = 1, \ldots, m. \tag{14}$$

In universal coding or statistics the corresponding quantity is usually called a *prior*.[22] It is normally used to improve robustness of probability estimators.

The value of parameter β that we found to be working well in the design of feature descriptors is computed as follows

$$\beta = \beta_0 \frac{n}{n_0}$$

where n_0 is the total number of samples in the original (non-quantized) histogram, and $\beta_0 = 1/2$ is the *prior* added in conversion of original histogram frequencies $k_{0,i}$ to probabilities:

$$p_i' = \frac{k_{0,i} + \beta_0}{n_0 + \beta_0 m}\,, \quad i = 1, \ldots, m.$$

This modification comes as gratis in terms of descriptor extraction complexity: we simply use Algorithm 1 to quantize unbiased input distribution $p_i = k_{0,i}/n_0$ to a point q in type lattice Q_n. As readily verified, this produces correct parameters $k_1, \ldots, k_m$ of a biased type (14).

4. FAST DECODING AND MATCHING

Recall, that in its final form, the compressed histogram of gradients descriptor is a sequence of codewords

$$\Xi = f\left(\xi_1\right) \ldots f\left(\xi_K\right)\,, \tag{15}$$

where K is the number of spatial bins, $\xi_i \in \{0, \ldots, |Q_n| - 1\}$ are indices of compressed distributions (types) in lattice Q_n, and $f : \{0, \ldots, |Q_n| - 1\} \to \{0,1\}^*$ is an encoding function.

When we use fixed-rate encoding, the function $f(.)$ simply outputs binary representations of indices ξ_i. Each codeword contains $R = \lceil \log |Q_n| \rceil$ bits. No particular decoding logic is needed in this case. When $f(.)$ denotes arithmetic encoding function, we follow the standard arithmetic decoding process to retrieve indices.

We next focus on conversion of type indices to distributions and computing distances between them. Let ξ_i^1 and ξ_i^2 denote type indices corresponding to i-th spatial cells of two feature descriptors Ξ^1 and Ξ^2. For example, let first descriptor be extracted from a query image, and the second descriptor extracted from an image in the database. In order to compute a distance between descriptors we now need to decode indices, convert them to distributions ($i = 1, \ldots, K$):

$$\xi_i^1 \rightsquigarrow \left\{k_{i,1}^1, \ldots, k_{i,m}^1\right\} \rightsquigarrow q_i^1 = \left[\tfrac{k_{i,1}^1 + \beta_i}{n_i + m\beta_i}, \ldots, \tfrac{k_{i,m}^1 + \beta_i}{n_i + m\beta_i}\right]\,,$$

$$\xi_i^2 \rightsquigarrow \left\{k_{i,1}^2, \ldots, k_{i,m}^2\right\} \rightsquigarrow q_i^2 = \left[\tfrac{k_{i,1}^2 + \beta_i}{n_i + m\beta_i}, \ldots, \tfrac{k_{i,m}^2 + \beta_i}{n_i + m\beta_i}\right]\,,$$

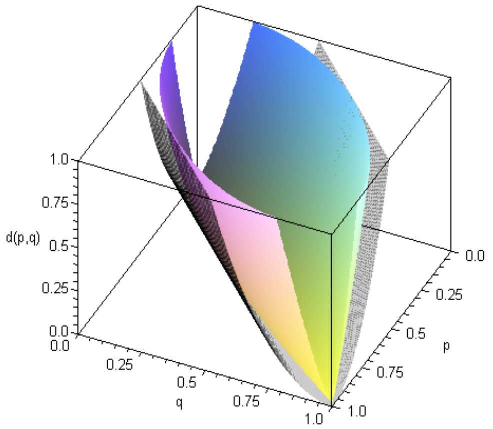


Figure 7. Pinsker-type inequality (19) in 2 dimensions. Symmetric KL-distance $d_\mathrm{J}(p,q)$ (shown in color) can be bounded by a function of $d_1(p,q)$ (shown in gray). This bound is tight when $p \to q$ or $p \to 1-q$.

and then compute a distance:

$$D(\Xi^1, \Xi^2) = \sum_{i=1}^{K} d(q_i^1, q_i^2)\,. \tag{16}$$

As noted in,[4,5] most of these operations can be avoided by precomputing a mapping:

$$d_\xi : \{0, \ldots, |Q_n| - 1\} \times \{0, \ldots, |Q_n| - 1\} \to \mathbb{R}_+$$

associating each pair of indices $\left(\xi_i^1, \xi_i^2\right)$ with a distance $d(q_i^1, q_i^2)$ between their reconstructed distributions. This enables *compressed-domain matching* of histogram-based descriptors.[4,5]

The size of the lookup table for compressed-domain matching is

$$M = |Q_n|^2 = \binom{n+m-1}{m-1}^2 .$$

When we use type lattices with parameter $n = O(m)$, the number of entries in such table becomes approximately:

$$M \sim \left(\frac{(1+\alpha)^{1+\alpha}}{\alpha^\alpha}\right)^{2m},$$

where $\alpha = n/m$. With small m and n this becomes a viable and remarkably fast solution.

When m or n is large, and/or matching needs to be done on a platform with very limited memory, a different technique is needed. Form Section 3.4 we know that decoding of type parameters (mapping $\xi_i^1 \rightsquigarrow \left\{k_1^1, \ldots, k_m^1\right\}$) can be done quickly by using formula (12). With precomputed binomial coefficients such process takes only $O(n)$ comparisons and additions. The computation of advanced distance measures, such as Kullback-Leibler or Jeffrey distances can be greatly simplified by using so-called *generalized Pinsker inequalities*:[26–28]

$$d_\mathrm{KL} \;\geqslant\; \frac{1}{2}\, d_1^{\,2}\,, \tag{17}$$

$$d_\mathrm{KL} \;\geqslant\; \log\frac{2+d_1}{2-d_1} - \frac{2d_1}{2+d_1}\,, \tag{18}$$

$$d_\mathrm{J} \;\geqslant\; d_1 \log\frac{2+d_1}{2-d_1}\,. \tag{19}$$

where d_KL denotes KL-distance (3), d_J is a symmetric form of KL-distance (4), and d_1 is an L_1-norm (also known as *variational distance* in this context[26]). We provide illustration of the last bound (19) in Figure 7.

The computation of the variational distance:

$$d_1(q_i^1, q_i^2) = \sum_{j=1}^{m} |q_{i,j}^1 - q_{i,j}^2| \,,$$

can also be simplified. For example, when q_i^1 and q_i^2 are obtained by using the same lattice parameter n_i and prior β_i, we just need to compute:

$$d_1(q_i^1, q_i^2) = \frac{1}{n_i + m\beta_i} \sum_{j=1}^{m} |k_{i,j}^1 - k_{i,j}^2| \,.$$

This formula operates directly with type coordinates $k_{i,1}^1, \ldots, k_{i,m}^1$, eliminating the need in conversion to probabilities, and allowing most operations to be performed in integer domain.

In summary, several options are available for fast decoding and matching of compressed histogram-based descriptors. Decoding and matching complexity exhibits at most linear $(O(K\,n))$ dependence on the number of spatial bins (K) and lattice parameter (n).

5. PERFORMANCE ANALYSIS

In this section we study performance of our quantization technique and histogram-based descriptors. We start with few results that can be produced analytically, and follow with experimental results verifying performance of our descriptors when working with real databases of images.

5.1 Analysis of Type Coding

We first note that vertices of Voronoi cells (or *holes*) in type lattice Q_m are located in positions

$$q_i^* = q + v_i, \quad q \in Q_n, \quad i = 1, \ldots, m-1 \,,$$

where $q \in Q_n$ are lattice points and v_i are vectors given by

$$v_i = \tfrac{1}{n}\Bigg[\underbrace{\tfrac{m-i}{m}, \ldots, \tfrac{m-i}{m}}_{i \text{ times}}, \underbrace{\tfrac{-i}{m}, \ldots, \tfrac{-i}{m}}_{m-i \text{ times}}\Bigg] \,.$$

This follows from analysis of geometry of Voronoi cells in lattice A_n (cf. [23, Chapter 21]).

The *covering radius* of type lattice Q_n can also be easily computed (as L_2-norm of vectors v_i):

$$\rho_2 \;=\; \max_{p \in \Omega_m} \min_{q \in Q_n} d_2(p,q) = \tfrac{1}{n}\sqrt{\tfrac{a(m-a)}{m}} \,, \qquad (20)$$

where $a = \lfloor m/2 \rfloor$. For L_∞ and L_1 norms we obtain:

$$\rho_\infty \;=\; \max_{p \in \Omega_m} \min_{q \in Q_n} d_\infty(p,q) \;\leqslant\; \tfrac{1}{n}\left(1 - \tfrac{1}{m}\right) \,, \qquad (21)$$

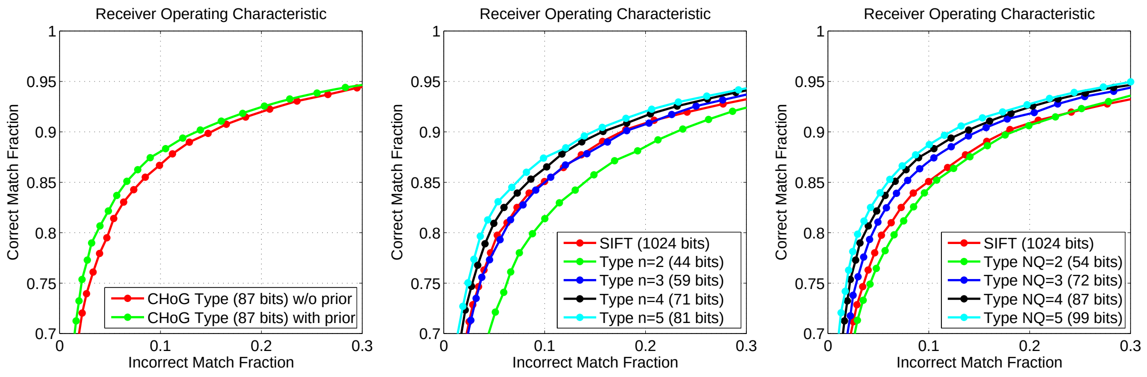
$$\rho_1 \;=\; \max_{p \in \Omega_m} \min_{q \in Q_n} d_1(p,q) \;\leqslant\; \tfrac{1}{n}\tfrac{2a(m-a)}{m} \,. \qquad (22)$$

We now establish connection with the coding rate of our quantizer (13):

$$R = \lceil \log_2 |Q_n| \rceil = (m-1)\log_2 n - \log_2 (m-1)! + O\left(\tfrac{1}{n}\right) \,.$$

This implies that (with $R \to \infty$):

$$n \sim 2^{\frac{R}{m-1}} \sqrt[m-1]{(m-1)!} \,,$$



(a) (b) (c)

Figure 8. Figure (a) shows the ROC curves of a type coded CHoG descriptor with and without priors. The performance of the descriptor is better with the scaled prior. Figure (b) shows ROC curves for compressing distributions with type coding scheme for DAISY-13 and VQ5 configuration for *Liberty* data set. Figure (c) shows ROC curves for compressing distributions with Type coding scheme for DAISY-13 and VQ7 configuration for *Liberty* data set. Larger number of gradient bins leads to better ROC performance.

and consequently (for example, by using (21)) that:

$$d^*_\infty[Q_n](\Omega_m, R) = \min_{n:|Q_n|\leqslant 2^R} \max_{p\in\Omega_m} \min_{q\in Q_n} d_\infty(p,q) \quad \lesssim \quad 2^{-\frac{R}{m-1}} \frac{1-\frac{1}{m}}{\sqrt[m-1]{(m-1)!}} . \tag{23}$$

Similar estimates are easily produced for other norms as well.

We observe, that the decay rate of $2^{-\frac{R}{m-1}}$ in (23) is exactly the same as predicted by the quantization theory (9). The only difference is in a leading constant factor. Thus, for L_∞ norm the factor in (9) becomes

$$\frac{1}{2}\sqrt[m-1]{\sqrt{m}} = \frac{1}{2} + O\left(\frac{\log m}{m}\right).$$

When using type lattice Q_n this factor is

$$\frac{1}{2} \leqslant 1 - \frac{1}{m} < 1,$$

which starts with $\frac{1}{2}$ when $m = 2$. This suggests that even when measured by the value of leading constant our algorithm comes close to the optimal performance.

Next, we evaluate performance of our compressed descriptors in the context of an image retrieval application.

5.2 Performance of Histogram-Based Image Feature Descriptors

To experimentally evaluate performance of our descriptors, we use the two data sets provided by Winder and Brown in their most recent work,[12] *Notre Dame* and *Liberty*. We extract descriptors from pixel patches around each interest point. For algorithms that require training, we use use the *Notre Dame* data set, while we perform our testing on the *Liberty* set.

We use the methodology proposed in Winder and Brown[12] for evaluating descriptors. We compute a distance between each matching and non-matching pair of descriptors. We use symmetric Kullback Leibler (KL) distance as the distance measure. From these distances, we compute a Receiver Operating Characteristic (ROC) curve which plots correct match fraction against incorrect match fraction. We compare our low bitrate descriptors to the well-known SIFT descriptor.[7]

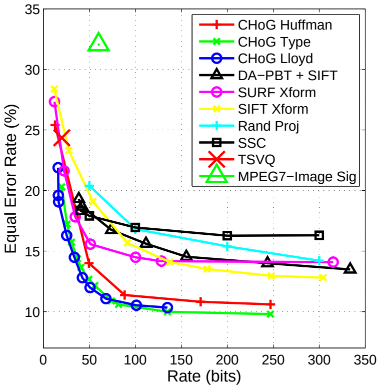


Figure 9. Comparison of EER versus bit-rate for all compression schemes for the *Liberty* data set. Better performance is indicated by a lower EER. We observe that CHoG outperforms all other schemes.

5.2.1 ROC Performance of CHoG Descriptor

We produce several ROC curves for our descriptors in Figure 8. Figure 8(a) illustrates the advantage of using biased types (14). These results were computed for DAISY-13, VQ-7 configuration, and with setting type parameter $n = 4$, and with $\beta_0 = 1/2$. Figure 8(b) shows performance of the type compression scheme for the DAISY-13, VQ-5 configuration. The bitrate in Figure 8(b) is varied by changing type quantization parameter n. For this configuration, the descriptor at 59 bits performs on par with SIFT at 1024 bits. Figure 8(c) shows ROC curves for compressing distributions with Type coding scheme for DAISY-13 and VQ-7 configuration for *Liberty* data set. It shows that larger number of gradient bins leads to better ROC performance, but at the expense of higher bitrates.

5.2.2 Comparison with Alternative Compression Schemes

In this section, we compare the performance of the different histogram compression schemes. For a fair comparison at the same bit rate, we consider the Equal Error Rate (EER) point on the different ROC curves for each scheme. The EER point is defined as the point on the ROC curve where the miss rate ($1 -$ correct match rate) and the incorrect match rate are equal. Figure 9 presents our EER results.

For more detailed descriptions of each scheme, please refer to.[4, 5]

- *CHoG with Huffman tree coding.* Instead of type-coding, this scheme quantizes distributions by constructing a Huffman tree, and then transmitting an index of this tree in a codebook of all possible Huffman trees for m-ary distributions. This idea was first used in the original design of CHoG descriptor.[4]
- *CHoG with generalized Lloyd coding.* Instead of type-coding, this scheme uses iterative Entropy Constrained Vector Quantization (ECVQ) procedure to produce a codebook and entropy codes that are optimally trained for a given image database.[5] This design is significantly more complex than type-coding.
- *Patch Compression.* We compress 32×32 pixel patches with DA-PBT (Direction Adaptive Partition Block Transform), which is shown to perform better than JPEG.[14] We compute a 128-dimensional 1024-bit SIFT descriptor on the reconstructed patch.
- *Random Projections.* Yeo *et al.*[15] propose the use of quantized random projections to build binary hashes from SIFT descriptors. We also compare CHoG to a machine learning algorithm called Boosting Similarity Sensitive Coding (Boost SSC)[16] which trains binary codes on SIFT descriptors to reflect patch similarity. Hamming distance between hashes is used as the distance measure.
- *Transform Coding.* Transform coding of SURF and SIFT descriptors was proposed by Chandrasekhar *et al.*[13] The compression pipeline first applies a Karhunen-Lòeve Transform (KLT) to decorrelate the different dimensions of the feature descriptor. This is followed by equal step size quantization of each dimension, and entropy coding with an arithmetic coder. We observe that CHoG descriptors outperform SIFT and SURF transform coding schemes at all bitrates.

- *Tree Structured Vector Quantization.* Here, we quantize SIFT descriptors by using a Vocabulary Tree[32] containing 1 million leaf nodes, thus requiring 20 bits per descriptor.
- *MPEG-7 Image Signatures.* As part of the MPEG-7 standard, Brasnett and Bober[17] propose a 60 bit signature for patches extracted around DoG interest points and Harris corners. The proposed method uses the Trace transform to compute a 1D representation of the image, from which a binary string is extracted using a Fourier transform. The descriptor is robust to simple image modifications like scaling, rotation, cropping and compression, but it is not robust to changes in perspective and other photometric distortions.[33]

Based on EER results presented Figure 9 we can observe that:

- CHoG descriptors designed with proposed type-quantization scheme perform very well, approaching best EER performance at rate of just 60 bits/descriptor,
- CHoG with Lloyd ECVQ quantizer, offers comparable or marginally better performance while being much more complex, and
- the other schemes perform significantly worse than CHoG in terms of EER vs. bitrate characteristics.

6. CONCLUSIONS

We have reviewed construction of a Compressed Histogram of Gradients (CHoG) image feature descriptor, and explained histogram quantization problem that arises in its design. We explained our choice of algorithms for solving it, focusing on both complexity and performance aspects. We also studied design of algorithms for decoding and matching of compressed descriptors, and offered several techniques for speeding up these operations. Proposed algorithms are memory-efficient, require small number of operations to execute, and well suitable for use in mobile phones, cameras, and other portable devices. Performance of CHoG descriptors with proposed quantization scheme is also studied. It is shown that CHoG descriptors achieve excellent equal-error-rate (EER) performance are very low data rates, significantly outperforming other known techniques.

REFERENCES

[1] *Google Goggles*, `http://www.google.com/mobile/goggles/`.

[2] *Nokia Point and Find*, `http://www.pointandfind.nokia.com`.

[3] *SnapTell*, `http://www.snaptell.com`.

[4] 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," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009).

[5] V. Chandrasekhar, Y. Reznik, G. Takacs, D. M. Chen, S. S. Tsai, R. Grzeszczuk, and B. Girod, "Quantization Schemes for Low Bitrate Compressed Histogram of Gradient Descriptors," *Proc. IEEE International Workshop on Mobile Vision (CVPR-IWMV'10)* (2010).

[6] G. Takacs, V. Chandrasekhar, N. Gelfand, Y. Xiong, W.-C. Chen, T. Bismpigiannis, R. Grzeszczuk, K. Pulli, and B. Girod, "Outdoors Augmented Reality on Mobile Phone using Loxel-Based Visual Feature Organization," *Proc. ACM International Conference on Multimedia Information Retrieval (MIR)* (2008).

[7] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, **60**, 2, 91–110 (2004).

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," *Proc. European Conference on Computer Vision (ECCV)* (2006).

[9] K. Mikolajczyk and C. Schmid, "Performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, 10, 1615–1630 (2005).

[10] Y. Ke and R. Sukthankar, "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors," *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, **02**, 506–513 (2004).

[11] G. Hua, M. Brown, and S. Winder, "Discriminant Embedding for Local Image Descriptors," *Proc. International Conference on Computer Vision (ICCV)* (2007).

[12] S. Winder, G. Hua, and M. Brown, "Picking the best daisy," *Proc. Computer Vision and Pattern Recognition (CVPR)* (2009).

[13] V. Chandrasekhar, G. Takacs, D. M. Chen, S. S. Tsai, and B. Girod, "Transform coding of feature descriptors," *Proc. Visual Communications and Image Processing Conference (VCIP)* (2009).

[14] M. Makar, C.-L. Chang, D.M. Chen, S.S. Tsai, and B. Girod, "Compression of image patches for local feature extraction," *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2009).

[15] C. Yeo, P. Ahammad, and K. Ramchandran, "Rate-efficient visual correspondences using random projections," *Proc. IEEE International Conference on Image Processing (ICIP)* (2008).

[16] G. Shakhnarovich and T. Darrell, "Learning Task-Specific Similarity," *Thesis* (2005).

[17] P. Brasnett and M. Z. Bober, "Robust visual identifier using the trace transform," *Proc. IET Visual Information Engineering Conference (VIE)* (2007).

[18] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2005).

[19] D. M. Y. Sommerville, *An Introduction to the Geometry of n Dimentions*, Dover, New York (1958).

[20] S. Graf, H. Luschgy, *Foundations of Quantization for Probability Distributions*, Springer-Verlag, Berlin (2000).

[21] I. Csiszár, "The method of types," *IEEE Transactions on Information Theory*, **44** (66) 2505–2523 (1998).

[22] T. M. Cover and J. M. Thomas, *Elements of Information Theory*, John Wiley & Sons, New York (2006).

[23] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*. Springer-Verlag, New York (1998).

[24] J. H. Conway and N. J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Transactions on Information Theory*, **28** (2) 227-232 (1982).

[25] Y.A. Reznik, "Quantization of Discrete Probability Distributions," *Proc. Workshop on Information Theoretic Methods in Science and Engineering (WITMSE)* (2010).

[26] M.S. Pinsker, *Information and Information Stability of Random Variables and Processes.*, Holden-Day, San Francisco (1964).

[27] I. Vajda, "Note on discrimination and variation," *IEEE Trans. on Information Theory*, **16**, 771-773 (1970).

[28] M. D. Reid, and R. C. Williamson, "Generalised Pinsker Inequalities," *Proc. 22nd Annual Conference on Learning Theory (COLT)* (2009).

[29] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," *Proc. International Workshop on Automatic Face and Gesture Recognition*, 296–301 (1994).

[30] E. Tola, V. Lepetit, and P. Fua, "A fast local descriptor for dense matching," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1–8 (2008).

[31] L. Van Gool, H. Shao, T. Svoboda, "Zubud-Zürich buildings database for image based recognition.," Tech. Rep. 260, ETH Zürich (2003).

[32] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2006)

[33] V. Chandrasekhar, A. L. Lin, G. Takacs, D. M. Chen, S. S. Tsai, N. M. Cheung, Y. Reznik, R. Grzeszczuk, and B. Girod, "Comparison of local feature descriptors for mobile visual search," *Proc. IEEE International Conference on Image Processing (ICIP)* (2010).