EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2009)
C. Grimm and J. J. LaViola Jr. (Editors)

A descriptor for large scale image retrieval based on sketched feature lines

Mathias Eitz[1], Kristian Hildebrand[1], Tamy Boubekeur[2] and Marc Alexa[1]

[1]Computer Graphics, TU Berlin, Germany
[2] Telecom ParisTech & LTCI CNRS, France

Abstract

We address the problem of large scale sketch based image retrieval, searching in a database of over a million images. The search is based on a descriptor that elegantly addresses the asymmetry between the binary user sketch on the one hand and the full color image on the other hand. The proposed descriptor is constructed such that both the full color image and the sketch undergo exactly the same preprocessing steps. We also design an adapted version of the descriptor proposed for MPEG-7 and compare their performance on a database of 1.5 million images. Best matching images are clustered based on color histograms, to offset the lacking color in the query. Overall, the query results demonstrate that the system allows users an intuitive access to large image databases.

Categories and Subject Descriptors (according to ACM CCS): H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing—Indexing methods

1. Introduction

Digital cameras have lead to vast amounts of digital images, many accessible for free through the internet (e.g. Flickr). Finding an image in a database that is close to a mental model is an important and difficult task. Currently, most queries are either based on textual annotations, rough color sketches or other images, respectively parts of images [SWS*00, DJLW08].

We feel that images cannot be succinctly communicated based on words; humans would probably describe different parts of the image and use different words depending on the cultural or professional background. On the other hand, searching an image based on a query that looks very similar to the intended result either requires an existing image, whose absence is usually the reason for a search, or great artistic skill if a shaded rendition of the image is necessary. It seems that it is much easier for humans to sketch the main feature lines of a shape or scene. This might be connected to how humans think of and memorize objects [KvDCL96, HS97, CGL*08]. Note that the main feature lines of an image almost completely determine its shading [Eld99]. This result has been exploited recently for creating a simple and intuitive to edit vector image representation [OBW*08].

The task of comparing a rough sketch of feature lines to an image is natural yet difficult. The first approaches to this problem go back to search based on pictorial description in 1979 [CF79]. Most approaches to image retrieval based on outline sketches up to now still use involved algorithms: Hirata *et al.* [HK92] search in a database of 205 colored oil-paintings by matching the edge image of the database images against the user sketch. Images are normalized in size and subdivided into 8×8 local blocks. For each local block, the best local correlation is computed by searching in a small window of local blocks. The global similarity is then computed as the sum of the local correlation values. Other similar methods are described in [KKOH92, CLLK97, RC00]. Lopresti *et al.* [LT95] recognized that a user sketch can be seen as a special form of handwriting and cleverly treat the search as a string matching problem in a database of 125 sketches. Jain *et al.* [JV96] combine color and shape information (using a linear combination of color histogram similarity and edge histogram similarity measures) to retrieve trademark images out of a database of 400 images.

Other works are based on matching a single curve to

the sketch: Del Bimbo *et al.* [DBP97] and Sclaroff [Scl97] let the user sketch undergo bend and stretch deformation to match the contours. Matusiak *et al.* [MDBA98] represent contours in curvature scale space [MM92] and define a distance measure for curves represented in curvature scale space. Ip *et al.* [ICWF01] present an affine invariant description for single contours. We believe that either deciding which single contour to extract or matching against a set of contours in each image is unlikely to scale to large databases.

Our main contribution is a sketch-based query system for image databases containing millions of images. As most current retrieval algorithms for large image databases it is based on a small descriptor that captures essential properties of the images. Typical descriptors use global or localized histograms of intensity, color, directionality [FSN*95, OT01, CBGM02, TFF08] or coefficients of global image transformations [JFS95, WWFW97]. These descriptors fail to generate good results for sketched feature lines as input. A descriptor for search based on edges [CNM05] employs an angular partitioning of the images and a histogram of the number of edge pixels falling into angular bins. The final feature vector is then computed as the Fourier transform of that histogram to achieve invariance to rotations. However, invariance to rotations also limits discrimination of local features.

We develop a new descriptor that is based on structure tensors [Knu89, KD08] (see Section 3). A main feature is that it elegantly addresses the asymmetry between the binary user sketch on the one hand and the full color image on the other hand. The proposed descriptor is constructed in such a way that both the full color image and the sketch undergo exactly the same preprocessing steps to compute the descriptor. This is new compared to several existing systems, gives an elegant formulation and considerably eases implementation.

We have also implemented an adapted version of the edge histogram descriptor proposed in the MPEG-7 standard [Sik01, YPJ*00] and use it to compare the performance on a database of 1.5 million images in Section 4.

The resulting sketch based image retrieval system can be used by any novice user to quickly query the image database (see Section 5 and the accompanying video). The power of the system stems from exploiting the vast amount of existing images, which offsets obvious deficits in image descriptors and search. We present more detailed conclusions in Section 5.

2. Overview

The input of our image search engine is a set of binary outlines which are sketched by the user to define the desired *shape* of the searched content and used to query the large image database. The result of this query is a small collection of pictures with similar structure but spanning a potentially large range of hues. In our system we typically query for approximately 50 to 100 images.

In order to provide the user with a mechanism for quickly finding the correctly colored image in the result set, we cluster the search results according to a color histogram descriptor into a small number of clusters (typically in the order of five to ten). Then the user can quickly find the cluster containing matches of desired color and choose from this cluster the image best matching the shape outlined in the sketch.

Our image ranking algorithm is based on descriptors which capture the main directions in each part of the image and are computed for all images in the database in an offline process. During the query, the user sketch provides direction information for each spatial region in the sketch and the descriptor generated from it is simply compared against all descriptors in the database. We analyze the properties of the proposed descriptor and evaluate its retrieval performance in Section 3.

3. An asymmetric sketch-vs-image descriptor

Almost all image descriptors are designed for matching entries in the database against a given (partial) image [SB91, FSN*95, JFS95, WWFW97, OT01, CBGM02, TFF08]. These descriptors can be used for user generated input only if this input resembles the image in color, intensity, or directionality. A vector-valued or scan-converted sketch of feature lines is not compatible with these descriptors, and we believe searching image databases based on this input can be considered harder than based on input already resembling the database entries.

Clearly, the main type of information in the input is the direction of the stroke (i.e. the tangents, resp. normals) relative to its position. This information relates best to the direction of gradients. Note that it is important to ignore the sign of the gradient, as the feature line only contains the information that gradients in the image are expected to be orthogonal to the line, but not which of the two regions is supposed to have higher intensity. In the following we discuss two descriptors that collect information about the gradients in each image in the database, which are specifically designed to be independent of the sign of gradients.

Let I denote an image with dimensions $m \times n$. We write $\mathbf{g}_{uv} = \nabla I_{uv}$ for the gradient. For both approaches we consider regular decomposition of the image into cells C_{ij}, typically between 24×16 and 32×24 per image. We say $(u,v) \in C_{ij}$ if the pixel with coordinates u and v is contained in the cell with index (i,j).

The main point of the descriptors described below is to determine the orientation of large gradients in each cell in the image hoping that they correlate with the normal directions of the user sketch. Note that the normals of the user sketch not only lack information on the sign but also have no "magnitude". This means we have to normalize the gradients or the descriptors, which results in regions with large and small gradients being treated equivalently. While we believe that

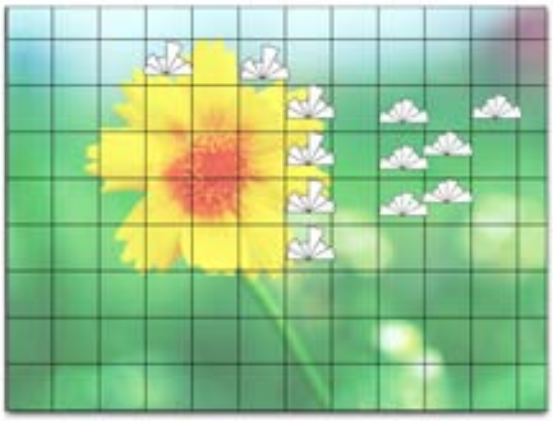

Figure 1: *The edge histogram descriptor stores for each image cell the sum of squared gradient magnitudes falling into one of six discrete orientation bins.*

indeed the prominence of a feature line has little to do with the gradient magnitude, we need to discard very small consistent gradients reflecting smooth intensity or color transitions or resulting from jpeg compression artifacts. In practice we set $\mathbf{g}^\mathsf{T}\mathbf{g} < \varepsilon^2$ to zero. We use $\varepsilon = \sqrt{2}/20$, which corresponds to 5% of the maximum gradient magnitude. We compute gradients on a grayscale image produced from the intensity channel of the input color image using the method of finite differences. When computing a descriptor from a binary image (user sketch), gradients are directly computed from the binary representation.

In order to retain pictures which contain an object fitting the user sketch but also other objects in different locations, every empty cell (i.e. which has no intersection with the user sketch) is ignored in the descriptor-based distance computation. This has three immediate consequences: first, the user can focus on specific picture content and does not have to sketch up an entire picture before querying the database; second, this increases the set of potentially acceptable results by avoiding restrictions on a picture's background; third, this reduces significantly the amount of distance computations during a query.

3.1. Edge histogram descriptor

We adapt a variant of the edge histogram descriptor (EHD) proposed in the MPEG-7 standard [Sik01, YPJ*00]. Another variant [DTRAM05] has been introduced for object recognition in images, as an alternative to shape contexts [BMP01] or SIFT [Low04]. It fits our requirement in that it only considers the gradients of the image and can easily be used without considering the sign of the gradient.

For each cell we compute gradient orientations and insert them into the corresponding histogram bin. We weigh each entry by its squared length based on the assumption that relatively stronger gradients are more likely to be sketched by the user. Let h_{ij} be the histogram of cell C_{ij} with d bins, then we define the weight in the k-th bin as

$$h_{ij}(k) = \sum_{(u,v)\in C_{ij}, o(\mathbf{g}_{ij})\in[k/d,k+1/d[} \mathbf{g}_{uv}^\mathsf{T}\mathbf{g}_{uv} \qquad (1)$$

with

$$o(\mathbf{x}) = \arccos\left(\operatorname{sgn}\left(\mathbf{e}^\mathsf{T}\mathbf{x}\right)\mathbf{e}^\mathsf{T}\mathbf{x}/\|\mathbf{x}\|\right) \qquad (2)$$

where $\mathbf{e}$ is an arbitrary unit direction vector and $\operatorname{sgn}\left(\mathbf{e}^\mathsf{T}\mathbf{x}\right)$ accounts for the desired equivalence $\mathbf{x} \equiv -\mathbf{x}$.

For the computation of distances between histograms we first compute normalized histograms H_{ij} to account for the possibly different number of gradients in two corresponding cells:

$$H_{ij} = \frac{1}{\sum_k h_{ij}(k)} h_{ij} \qquad (3)$$

Now let H_{ij} and $\tilde{H}_{ij}$ denote two normalized histograms. Let d_{ij} denote the L_1 distance between H_{ij} and $\tilde{H}_{ij}$:

$$d_{ij} = \sum_k |H_{ij}(k) - \tilde{H}_{ij}(k)| \qquad (4)$$

We now can define the distance between two edge histogram descriptors H and $\tilde{H}$ as:

$$\operatorname{dist}(H,\tilde{H}) = \sum_i \sum_j d_{ij} \qquad (5)$$

The resulting image description is visualized using pie charts in Figure 1.

3.2. Tensor descriptor

Contrary to the histogram approach where orientations are discretized into bins the structure tensor gives us information about the main orientation of the gradients in a cell. In particular, we are interested in finding a single vector in a cell that is *as parallel as possible* to the image gradients in that cell. This vector would be a representative for the image "structure" in that cell.

We pose this as a maximization problem and see that the system matrix corresponds to the so-called structure tensor. We only consider discrete scalar images containing luminances here, while the approach can be easily extended for multi-band images [DZ86].

Let $\mathbf{x}$ be a unit vector, which we want to define such that it represents the main direction in cell C_{ij}. As $\mathbf{x}^\mathsf{T}\mathbf{g}_{uv}$ attains a maximum if $\mathbf{x} \parallel \mathbf{g}_{uv}$ we pose the definition of $\mathbf{x}$ as the following optimization

$$\mathbf{x} = \underset{\|\mathbf{x}\|=1}{\arg\max} \sum_{(u,v)\in C_{ij}} \left(\mathbf{x}^\mathsf{T}\mathbf{g}_{uv}\right)^2. \qquad (6)$$

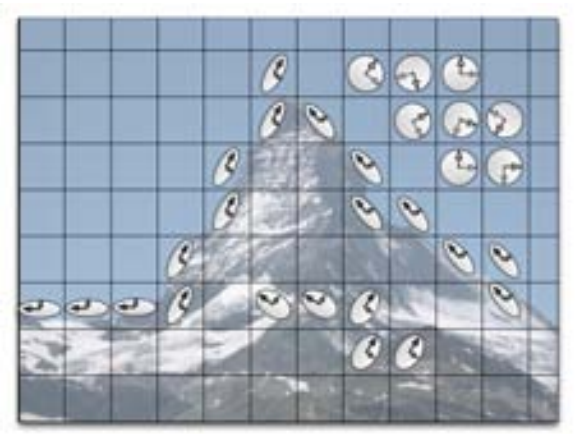

Figure 2: *The tensor descriptor subdivides the image into rectangular tiles. For each tile a structure tensor is computed, depicted by the ellipses.*

Note that

$$\sum_{(u,v)\in C_{ij}} \left(\mathbf{x}^\mathsf{T}\mathbf{g}_{uv}\right)^2 = \sum_{(u,v)\in C_{ij}} \mathbf{x}^\mathsf{T}\mathbf{g}_{uv}\mathbf{g}_{uv}^\mathsf{T}\mathbf{x} = \mathbf{x}^\mathsf{T}\left(\sum_{(u,v)\in C_{ij}} \mathbf{g}_{uv}\mathbf{g}_{uv}^\mathsf{T}\right)\mathbf{x} = \mathbf{x}^\mathsf{T} G_{ij}\mathbf{x} \tag{7}$$

which means we are maximizing a quadratic function in $\mathbf{x}$ with the constraint $\mathbf{x}^\mathsf{T}\mathbf{x} = 1$. The matrix G_{ij} contains the sum of outer products of gradients in cell C_{ij} and is commonly referred to as the structure tensor. We find the unique maximum using the Langrange multiplier λ, and setting $\nabla\mathbf{x}$ to zero leads to the necessary condition

$$2G_{ij}\mathbf{x} + 2\lambda\mathbf{x} = 0 \tag{8}$$

which means that we can find $\mathbf{x}$ (up to sign) as the unit eigenvector of G_{ij} corresponding to the largest eigenvalue. The eigenvalues correspond to the maximum and minimum of the quadratic functional, reflecting the distribution of gradients. Thus, a compact representation of all this information, yet not including the sign of the gradients, is given by the structure tensor G_{ij}.

In order to detect similarly oriented image edges independently of the magnitude of the edges, we store the structure tensor normalized by its Frobenius norm:

$$T_{ij} = \frac{G_{ij}}{\|G_{ij}\|_F} \tag{9}$$

We define the distance d_{ij} between two tensors T_{ij} and $\tilde{T}_{ij}$ as the Frobenius Norm of the difference between the two tensors:

$$d_{ij} = \|T_{ij} - \tilde{T}_{ij}\|_F \tag{10}$$

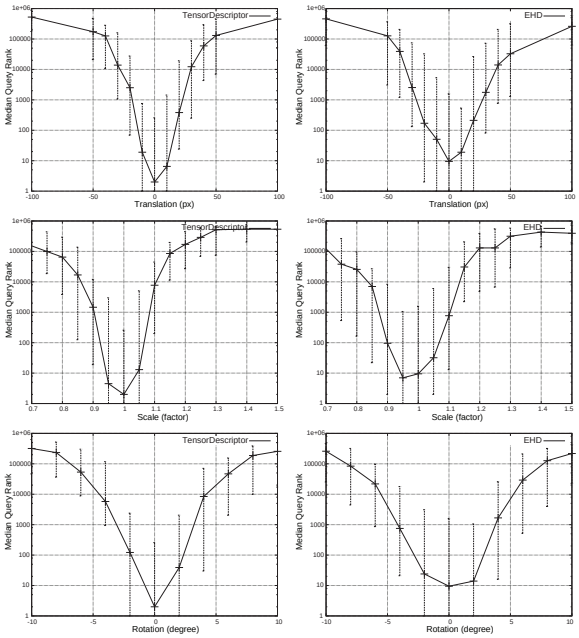

Figure 3: *The results of an objective evaluation of the image descriptors. For 27 images contained in the database of 1.5 million images we have generated queries based on hand drawn sketches. The graphs show the median rank and the upper and lower quartile of the original image over translations, rotations, and scales of the input.*

Finally, we define the distance between two tensor descriptors as the sum over the tensor distances in their corresponding cells:

$$\mathrm{dist}(T,\tilde{T}) = \sum_i \sum_j d_{ij} \tag{11}$$

A visualization of the resulting image descriptor is given in Figure 2.

3.3. Descriptors performance

We have studied the relative performance of the edge histogram descriptor and the tensor descriptor. We have selected 27 reference images from our image database and generated 27 corresponding binary outline sketches, similar to those a user would sketch in the interactive version of the software. Those sketches have been created by three different users that were instructed to follow the most important outlines in the image. Some of these sketches can be seen in Figure 6. Using this set of reference sketches we have evaluated descriptor performance by querying the database for the most similar images to each sketch and finding the rank of the reference image in the resulting answer. To check the robustness of the descriptors we generated queries for translated, scaled, and rotated versions of the input sketch. Fig-

ure 3 summarizes the results of this evaluation graphically. As expected, images are most likely recovered from a sketch if the sketch is in the right position, scale, and orientation, however, we see that small amounts of transformation are tolerable.

However, we also found that the rank of the sketched image is of only limited value in the context of very large databases and also our requirements. A good descriptor is not supposed to discriminate objects belonging to the same class. If many objects similar to the sketch are contained in the database it is unlikely that the source image is found. We demonstrate this effect for a query that has essentially not been successful, i.e. the rank of the original image is very large. Figure 4 shows the 15 images with smallest distance to the sketch input shown (using the tensor descriptor). While the original image is not among these 15 images (in fact the rank in this case is 5474) most images in this set clearly show a resemblance to the input.

While in the objective evaluation we can recognize a slight advantage of the tensor descriptor over the edge histogram descriptor, the users of our experimental system indeed seemed to prefer the results generated by the tensor descriptor. We thus used the tensor descriptor in all our examples, unless stated otherwise.

4. Database, image search and clustering

We have downloaded a set of 1.5 million pictures retrieved from various Flickr groups, all related to outdoor sceneries. We have made sure that all images have an aspect ratio of 4:3, cropping images that had other aspect ratios. Only images with a minimum resolution of 640x480 have been retrieved; additionally the maximum size has been limited to 1024x768 pixels, downscaling larger images. All downloaded images have been stored in jpeg format in a simple folder structure on harddisk. The database memory footprint is 375GB resulting in an average jpeg filesize of 250Kb.

We store the images and run the server providing the search service on a standard Apple MacPro configured with 2 Intel Xeon 2.8Ghz QuadCore processors and 32GB of main memory. The server preloads and maintains an array of all image descriptors. We have chosen the parameters so that both the edge histogram descriptor and the tensor descriptor take exactly 9Kb memory per image. When starting the system we load all descriptors into a linear array which is kept in main memory. A query is performed by linear search and selecting the k results with smallest distance to the input descriptors (typically generated from a sketch). We use a fixed-size priority queue of k elements to retain the query result. We have found that linear search is fast enough for querying the database interactively in our prototype system. Note that improving the search is non-trivial due to the high dimensionality of the entries [BGRS99].

Before presenting a query result to the user we cluster the result set (typically 50 to 100 images) into 5 to 10 clusters of similarly colored images using the k-means algorithm. We measure color distance between images as the L_1 distance between their corresponding color histograms. We employ a three-dimensional color histogram, subdividing the RGB colorspace into $6 \times 6 \times 6$ bins. We show a result of clustering a query in Figure 5.

5. Results and conclusion

We have developed a tensor based image descriptor for *large scale* sketch based image retrieval. We have shown that the descriptor's performance is superior to a variant of the MPEG-7 edge histogram descriptor in a quantitative evaluation for which we have measured retrieval ranks of 27 sketches created from reference images out of the image database. The results of the evaluation are shown in Figure 3.

As can be seen in Figures 4,5,6 the tensor descriptor gathers good matches for a given query sketch. We show a typical result of a query in Figure 4, displaying the *first* 15 matches. In Figure 6 we show a hand-picked subset of the top 100 matches for each of the three query sketches. For each sketch we show six images that match the probably intended semantics of the sketch and are considered good matches by the experimental users of our system. While the intended semantics of a sketch is not reflected in all of the answers, they still resemble the features in the user sketch; this is shown in each second row of Figure 6.

The proposed descriptor works well and performs comparably or slightly better than the MPEG-7 edge histogram descriptor variant. It is easy to implement and efficient in evaluation – a query in the 1.5 million image database takes between 0.4 and 3.5 seconds depending on the sparsity of the user sketch. When the user sketch is sparse, many cells get masked out and the number of distance calculations per image is reduced, resulting in a faster query. Moreover, the descriptor's computation is very fast: we process about 70 images per second on our 8 Core desktop machine when computing descriptors in the offline process.

Additionally, we have implemented a simple user interface for drawing and editing sketches that allows to interactively query our database of 1.5 million images. While we have not performed any user study, the use of simple outlines for querying the database has been intuitive to use in our experiments. We show user interaction with the system in the accompanying video.

The system could be improved by reducing the memory footprint (exactly 9Kb) of the image descriptors by using e.g. quantization or learning a compact binary code such that the pairwise descriptor distances are conserved [TFW08]. Out-of-core search would allow running it on smaller machines with limited main memory. While for our database size (1.5 million images) the performance of the linear search was not a limitation, larger databases could certainly make use of

Figure 4: *The first 15 matches (left to right, top to bottom) for the query on the left. The query sketch had been generated from an image in the data base, which has been ranked 5474. Note that the first matches provide a very reasonable answer to the user query, meaning that a low rank of the image used to generate the sketch does not imply the descriptor failed.*

Figure 5: *Answer of the proposed system to the sketch shown in the middle. The first* 50 *matches are clustered into* 6 *clusters. Note that the result set contains a very high percentage of trees as probably desired by the user.*

faster searches, employing e.g. approximate nearest neighbor techniques [IM98].

While our descriptor can be efficiently computed and evaluated, it provides only limited invariance to similarity transformations (see Figure 3). We believe that such deficits in a descriptor can be overcome by exploiting the variety provided by a *large* image database and support this claim with the results shown in Figures 4,5,6 and the accompanying video.

An interesting observation is the dependence of the system on the database content. We have tried to gather only "good" images for our database. As a result the database contains relatively few objects in a simple frontal view (i.e. the front side of the house, the side view of a car). However, most users tend to sketch objects from these points of view and will find that only few images match their sketch – simply because there are no objects in the database with silhouettes as sketched by the user.

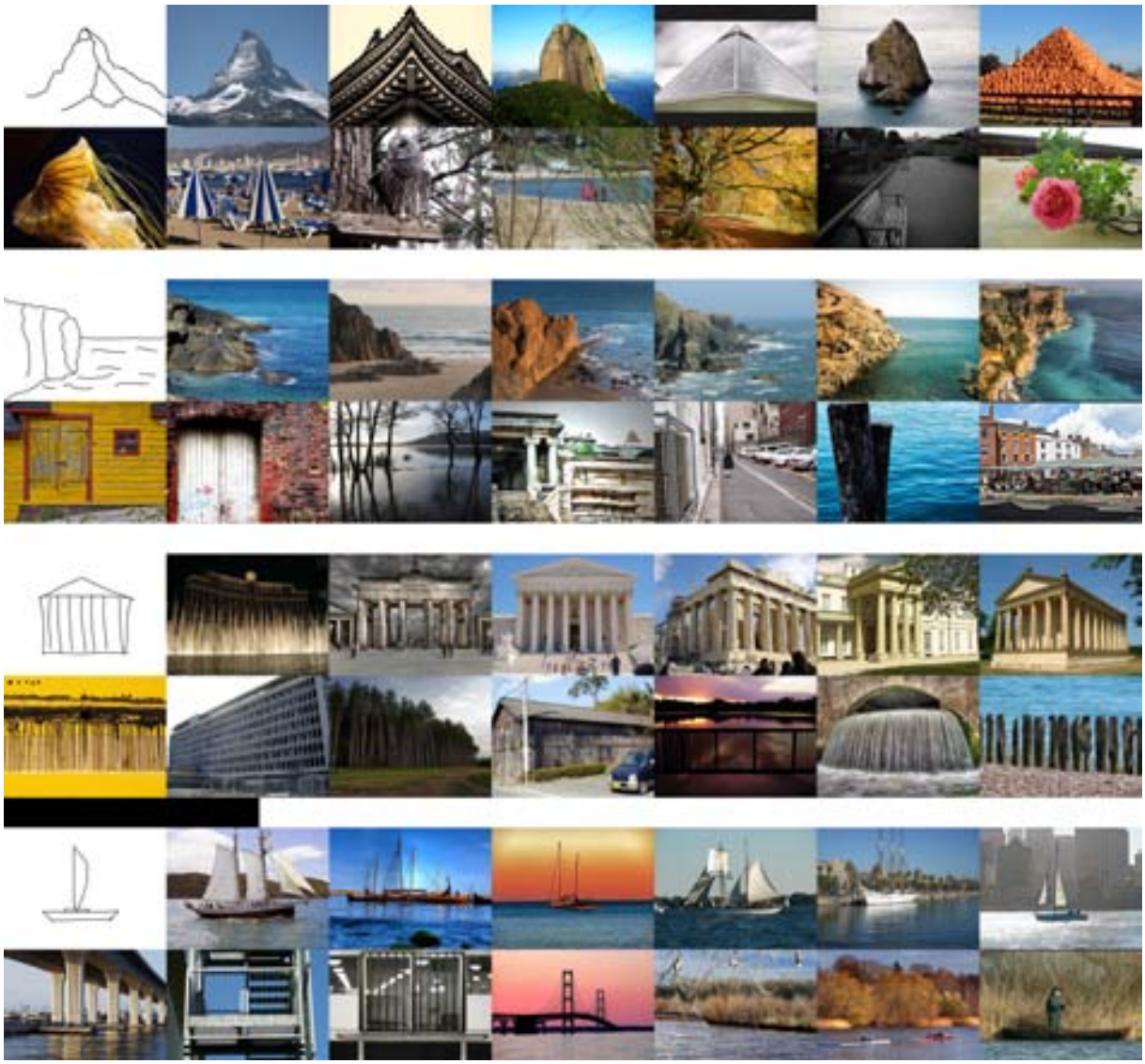

Figure 6: *Shown is a hand-picked subset of the results when querying the database for 50 images matching the sketches on the top left of a row. In the top row we show "expected" results, and rather "unexpected" results in the corresponding bottom row.*

References

[BGRS99] BEYER K., GOLDSTEIN J., RAMAKRISHNAN R., SHAFT U.: When Is" Nearest Neighbor" Meaningful? *LECTURE NOTES IN COMPUTER SCIENCE* (1999), 217–235.

[BMP01] BELONGIE S., MALIK J., PUZICHA J.: Matching shapes. In *ICCV* (2001), pp. 454–463.

[CBGM02] CARSON C., BELONGIE S., GREENSPAN H., MALIK J.: Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 8 (2002), 1026–1038.

[CF79] CHANG N., FU K.: Query-by-pictorial-example. In *The IEEE Computer Society's Third International Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79* (1979), pp. 325–330.

[CGL*08] COLE F., GOLOVINSKIY A., LIMPAECHER A., BARROS H. S., FINKELSTEIN A., FUNKHOUSER T., RUSINKIEWICZ S.: Where do people draw lines? *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 88:1–88:11.

[CLLK97] CHAN Y., LEI Z., LOPRESTI D., KUNG S.: A feature-based approach for image retrieval by sketch. *SPIE Storage and Retrieval for Image and Video Databases II* (1997).

[CNM05] CHALECHALE A., NAGHDY G., MERTINS A.: Sketch-based image matching using angular partitioning. *IEEE Transactions on Systems, Man and Cybernetics, Part A 35*, 1 (2005), 28–41.

[DBP97] DEL BIMBO A., PALA P.: Visual image retrieval by elastic matching of user sketches. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 2 (1997), 121–132.

[DJLW08] DATTA R., JOSHI D., LI J., WANG J. Z.: Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys 40*, 2 (2008), 1–60.

[DTRAM05] DALAI N., TRIGGS B., RHONE-ALPS I., MONTBONNOT F.: Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 1.

[DZ86] DI ZENZO S.: A note on the gradient of a multi-image. *Computer Vision, Graphics, and Image Processing 33*, 1 (1986), 116–125.

[Eld99] ELDER J.: Are edges incomplete? *International Journal of Computer Vision 34*, 2 (1999), 97–122.

[FSN*95] FLICKNER M., SAWHNEY H., NIBLACK W., ASHLEY J., HUANG Q., DOM B., GORKANI M., HAFNER J., LEE D., PETKOVIC D., STEELE D., YANKER P.: Query by image and video content: The QBIC system. *IEEE Computer 28*, 9 (Sept. 1995), 23–32.

[HK92] HIRATA K., KATO T.: Query by visual example-content based image retrieval. In *Proceedings of the 3rd International Conference on Extending Database Technology: Advances in Database Technology* (1992), Springer-Verlag London, UK, pp. 56–71.

[HS97] HOFFMAN D. D., SINGH M.: Salience of visual parts. *Cognition*, 63 (1997), 29–78.

[ICWF01] IP H., CHENG A., WONG W., FENG J.: Affine-invariant sketch-based retrieval of images. *Computer Graphics International* (2001).

[IM98] INDYK P., MOTWANI R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (1998), ACM New York, NY, USA, pp. 604–613.

[JFS95] JACOBS C. E., FINKELSTEIN A., SALESIN D. H.: Fast multiresolution image querying. In *Proceedings of SIGGRAPH 95* (Aug. 1995), pp. 277–286.

[JV96] JAIN A., VAILAYA A.: Image retrieval using color and shape. *Pattern Recognition 29*, 8 (1996), 1233–1244.

[KD08] KYPRIANIDIS J. E., DÖLLNER J.: Image abstraction by structure adaptive filtering. In *Proc. EG UK Theory and Practice of Computer Graphics* (2008), pp. 51–58.

[KKOH92] KATO T., KURITA T., OTSU N., HIRATA K.: A sketch retrieval method for full color image database-query by visual example. *Pattern Recognition* (1992).

[Knu89] KNUTSSON H.: Representing local structure using tensors. In *The 6th Scandinavian Conference on Image Analysis* (June 1989), Oulu, Finland, pp. 244–251.

[KvDCL96] KOENDERINK J. J., VAN DOORN A. J., CHRISTOU C., LAPPIN J. S.: Shape constancy in pictorial relief. In *Object Representation in Computer Vision II* (1996), p. 151.

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (Nov. 2004), 91–110.

[LT95] LOPRESTI D., TOMKINS A.: Temporal domain matching of hand-drawn pictorial queries. In *Proc. of the Seventh Conf. of The Intl. Graphonomics Society* (1995), pp. 98–99.

[MDBA98] MATUSIAK S., DAOUDI M., BLU T., AVARO O.: Sketch-based images database retrieval. *Lecture notes in computer science* (1998), 185–191.

[MM92] MOKHTARIAN F., MACKWORTH A.: A theory of multiscale, curvature-based shape representation for planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 8 (1992), 789–805.

[OBW*08] ORZAN A., BOUSSEAU A., WINNEMÖLLER H., BARLA P., THOLLOT J., SALESIN D.: Diffusion curves: a vector representation for smooth-shaded images. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)* (2008), vol. 27, pp. 1–8.

[OT01] OLIVA A., TORRALBA A.: Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision 42*, 3 (2001), 145–175.

[RC00] RAJENDRAN K., CHANG R.: Image retrieval with sketches and compositions. *2000 IEEE International Conference on Multimedia and Expo* (2000).

[SB91] SWAIN M., BALLARD D.: Color indexing. *International Journal of Computer Vision 7*, 1 (1991), 11–32.

[Scl97] SCLAROFF S.: Deformable prototypes for encoding shape categories in image databases. *Pattern Recognition 30*, 4 (1997), 627–641.

[Sik01] SIKORA T.: The MPEG-7 Visual standard for content description-an overview. *IEEE Transactions on Circuits and Systems for Video Technology 11*, 6 (2001), 696–702.

[SWS*00] SMEULDERS A., WORRING M., SANTINI S., GUPTA A., JAIN R.: Content-based image retrieval at the end of the early years. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*, 12 (2000), 1349–1380.

[TFF08] TORRALBA A., FERGUS R., FREEMAN W. T.: 80 million tiny images: a large database for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 11 (November 2008), 1958–1970.

[TFW08] TORRALBA A., FERGUS R., WEISS Y.: Small codes and large image databases for recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), pp. 1–8.

[WWFW97] WANG J. Z., WIEDERHOLD G., FIRSCHEIN O., WEI S. X.: Content-based image indexing and searching using daubechies' wavelets. *Int. J. on Digital Libraries 1*, 4 (1997), 311–328.

[YPJ*00] YAMADA A., PICKERING M., JEANNIN S., CIEPLINSKI L., OHM J.-R., EDITORS M.: Mpeg-7 visual part of experimentation model version 8.0. *ISO/IEC JTC1/SC29/WG11/N3673* (Dec 2000), 1–82.