PixSearcher: Searching Similar Images in Large Image Collections through Pixel Descriptors

Tuan Nhon Dang and Leland Wilkinson

University of Illinois at Chicago

Abstract. Searching and mining huge image databases has become a daunting task for many application areas such as astronomy, medicine, geology, oceanography, and crime prevention. We introduce a new image indexing and retrieval system, PixSearcher, that computes image descriptors from pixels obtained by thresholding images at different levels. In contrast to raster techniques (which use the entire pixel raster for distance computations), PixSearcher uses a small set of descriptors and consequently can handle large image collections within reasonable time. We use benchmark image databases from different domains to evaluate PixSearcher's performance versus well-known image retrieval techniques.

1 Introduction

With the rapid growth of the Internet, storage devices, and digital photographs, image retrieval has become increasingly popular in the past twenty years. One of the most popular image search techniques [1] is based on human-supplied text annotations to describe image semantics (concept-based approach). Text searching algorithms are then used to obtain similar images. However, there are problems with this technique. Different people may tag the same image differently due to emotive cues, context, and culture. Moreover, text annotations are not always available. There are recent efforts [2] to generate text annotations automatically, but evaluating the accuracy of this process is as difficult as evaluating the accuracy of image retrieval itself.

To overcome the drawbacks of text-based image retrieval, content-based images retrieval (CBIR) was introduced. In contrast to concept-based approaches, content-based approaches analyze images based on colors, shapes, textures in the images rather their annotations. In general, CBIR is more efficient, more accurate, and less expensive. Our approach belongs to this class.

This work is a natural extension of the work on *Scagnostics* [3]. Scagnostics characterizes the shape of 2D scatterplots by operating on descriptors of point distributions. These characterizations include measures, such as clumpiness, stringiness, association, and density. Although the motivation for Scagnostics was to locate interesting scatterplots in a large scatterplot matrix, we realized the idea had more general implications. In this paper, we extend this idea to handle pixels. Our algorithm involves several stages. We first rescale images. We then quantize these rescaled images on three channels (Hue, Saturation, and

Brightness) and compute descriptors for the configurations of these quantized pixels. Finally, similarity of images is measured based on their descriptors.

2 Related Work

The number of image retrieval techniques has been increasing in the past decade. Some image retrieval techniques focus on a specific domain. Other image retrieval techniques work on general-purpose images. The general-purpose CBIR systems share one common procedure: descriptors are extracted from the query image. These descriptors are then compared to the images in a database using a selected distance measure. Finally, the most relevant image is returned. The accuracy of a CBIR system depends on how well these features characterize the target images and how suitable the distance measure is selected for these features.

The general-purpose CBIR systems can be roughly classified into three categories: color histograms, region-based techniques, and feature-based techniques.

2.1 Color Histogram

Color histogram is a fundamental and commonly used technique [4]. The basic idea behind color histogram is to learn the color distributions of images. To compare two color histograms, Euclidean distance is the most commonly used. Color histogram often use Red, Green, and Blue color space for quantizing process.

2.2 Region-Based techniques

A region-based retrieval system decomposes an image into regions using image segmentation [5]. Therefore, the accuracy of region-based techniques depends on the accuracy of automatic image segmentation. The techniques belonging to this category include NeTra [6], Blobworld [7]. These techniques present a user with the segmented regions of an image. The user selects regions and attributes to be matched. SIMPLIcity [8] is able to match segmented regions automatically using integrated region matching (integrated region matching tries to incorporate two region sets to compute similarity scores).

Another approach in this category [9] uses the dissimilarity function of Windsurf [10] for multi-region queries. Windsurf formulates the matching between two region sets as an optimization problem. Using efficient multi-step algorithms instead of a brute-force search enables this approach to work on larger image databases. Experiments on a database with more than 370,000 images have shown the multi-step approach outperformed the previous approaches.

2.3 Feature-based techniques

The MPEG-7 standard includes color descriptors (such as the dominant color descriptor) and texture descriptors (such as the edge histogram texture descriptor). These feature are computationally inexpensive. The best combination of

these descriptors is color layout, dominant color, edge histogram, and texture browsing [11].

SIFT (Scale Invariant Feature Transform) features are invariant to image scaling, translation, and rotation [12]. These features are used for object recognition. The recognition accuracy can be improved by incorporating other image aspects such as color and texture. Bag of Words [13] extracts local features from the images, e.g. SIFT. Visual words are then generated by clustering these local features. Images are represented as histogram counts of visual words which are then used for image comparisons.

Gabor features [14] are inspired by text retrieval research. This approach uses an inverted file with more than 8,000 features per image. Weighting scheme in text retrieval is applied for frequency of occurrence of features. Relevant feedback is also used to improve the performance (precision and recall) of this approach.

3 PixSearcher

3.1 PixSearcher Design

Figure 1 shows a schematic overview of our image retrieval approach. We design our system as close as possible to the way the human perceive objects: for every stage in the bottom-up process [15], there is a corresponding stage in our approach. In particular, a query image is preprocessed (rescaled and then quantized) to create low-level representation of the query image. Image descriptors are then computed on the quantized pixels. These descriptors collectively represent the query image. The set of features of the query image is compared to images in the training image dataset. Finally, PixSearcher outputs the most similar image in the training dataset.

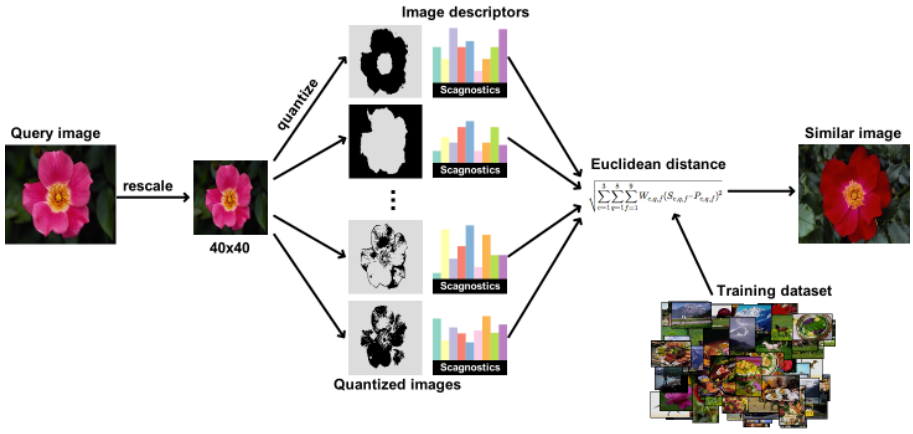

Fig. 1. Schematic overview of PixSearcher.

Humans are the users of CBIR systems who are able to access the validity of the output. Therefore, we design our image descriptors based on visual perception theory. We now outline our image algorithms.

3.2 Quantizing Images

We begin by rescaling the query image into 40 by 40 pixel arrays. The choice of rescaling size is constrained by efficiency (too many pixels slow down calculations) and sensitivity (too few pixels obscure features in the images). We map each pixel $p_{ij}; i = 1;:::;40; j = 1;:::;40$ to the tuple $(x_k; y_k); k = 1;:::;1600$, where $0 \square x_k \square 1$ and $0 \square y_k \square 1$.

Next, we quantize the 40 by 40 pixel images on Hue, Saturation, and Brightness (HSB). For each of these color dimensions, we produce q binary pixel arrays (the selection of q will be justified later in Section 4.1). In each of these arrays, a pixel takes the value 1 if its color dimension lies within a quantile and 0 otherwise. We then restrict our tuples to the set X containing the tuples corresponding to quantized pixels with value 1.

Figure 2 shows an example of quantization. In particular, the rescaled version of the input image (the logo of visitnorway.com) is on the top. The three image rows display quantization results of the rescaled image on Hue, Saturation, and then Brightness. On each color channel, we apply 5 quantizations (0 to 0.2, 0.2 to 0.4, 0.4 to 0.6, 0.6 to 0.8, and 0.8 to 1). The black dots in the results are the quantized pixels. On Hue, we can extract different color components. The last two quantized images on Brightness reveal different meanings: the logo of visitnorway.com can be viewed as both three separate leaves and a person stretching his/her arms into the air. This example also helps to explain the methodology of our approach: we first extract different components in the input image by quantizing and then learn the semantics of these components independently.

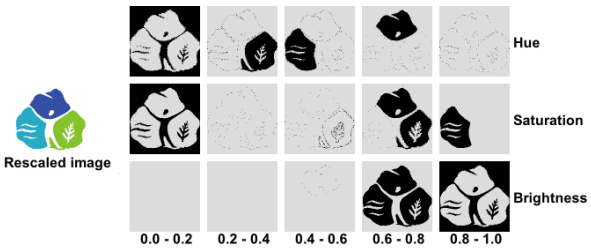

Fig. 2. Quantization results on HSB of a sample image.

3.3 Computing image descriptors

We select suitable scagnostics features [3] and adapt them to the image context. These extended features are Close, Dense, Convex, Skinny, and Clumpy. We

now outline the computation of the descriptors. These descriptors are computed based on proximity graphs that are subsets of the Delaunay triangulation. In the formulas below, we use H for the convex hull, A for the alpha hull, and T for the minimum spanning tree (MST). Each of these proximity graphs is computed on the set X of quantized image.

1. **Pixel level**: To compute the Pixel level feature, we first count the number of pixels satisfying the filtering conditions, namely, $|X|$. This number is then normalized by the maximum number of pixels in the rescaled image.

$$f_{pixel\_level} = |X|/(40 \times 40) \tag{1}$$

2. **Close**: Our Close descriptor is derived from the Outlying scagnostic which is used to detect the data points that are notably separated from the remaining vertices in the MST. As opposed to the Outlying scagnostic, the Close descriptor measures the closeness of pixels in the quantized image. The Close descriptor is computed based on the proportion of the total edge length of the MST accounted for by the total length of edges connecting adjacent quantized pixels (edges of length 1).

$$f_{close} = length(T_1)/length(T) \tag{2}$$

3. **Dense**: Our Dense descriptor compares the area of the alpha shape to the area of the whole frame. Low values of this statistic indicate a sparse image. This descriptor addresses the question of how fully the quantized pixels fill the frame. Examples of the Close and Dense features are given in Figure 3.

$$f_{dense} = area(A)/(40 \times 40) \tag{3}$$

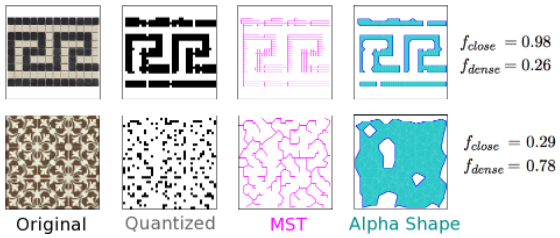

Fig. 3. Top image is high Close and sparse. Bottom image is low Close and dense.

4. **Convex**: Our convexity measure is based on the ratio of the area of the alpha hull and the area of the convex hull. This ratio will be 1 if the alpha hull and the convex hull have identical areas.

$$f_{convex} = area(A)/area(H) \tag{4}$$

5. **Skinny**: The ratio of perimeter to area of a polygon measures, roughly, how skinny it is. We use a corrected and normalized ratio so that a circle yields

a value of 0, a square yields 0.12 and a skinny polygon yields a value near one. Examples of the Convex and Skinny features are given in Figure 4.

$$f_{skinny} = 1 - \sqrt{4 \cdot area(A)/perimeter(A)} \quad (5)$$

Fig. 4. Top image is Convex and low Skinny. Bottom image is low Convex and Skinny.

6. Clumpy: The Clumpy feature aims to detect clusters of data points. Different from the Clumpy scagnostic (which is based on the RUNT statistic [16]), our Clumpy image descriptor is computed based on the number of alpha shapes $|A|$. Examples of the Clumpy features are given in Figure 5.

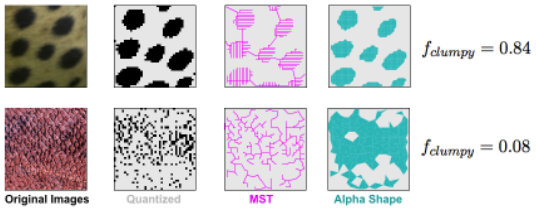

Fig. 5. Top image is high Clumpy. Bottom image is low Clumpy.

We apply 8 quantizations for each color dimension. Then, we compute 6 descriptors for each quantized image. Totally, each image is characterized by 144 descriptors. The dissimilarity of two images is computed by using Euclidean distance on their feature space.

4 Testing

We use 4 benchmark image databases from different domains for evaluation. They are described in detail in a survey paper [17]. We will use the evaluation results from this paper to compare to PixSearcher's performance.

4.1 Selection of number of quantizations

How many quantizations to apply on each color channel? Figure 6 shows the test results of PixSearcher with different quantizations on the 4 image datasets. In particular, error rate test results on the left of Figure 6 indicate that PixSearcher does not perform well with fewer quantizations. As we increase the number of quantizations, the error rates get smaller. The significant improvement is obtained with 8 quantizations (the smallest average error rates over four datasets). The results get worse if we keep increasing the number of quantizations. This is because we do not have enough pixels in each quantized image to characterize different aspects of an image. Therefore, we use 8 quantizations for PixSearcher.

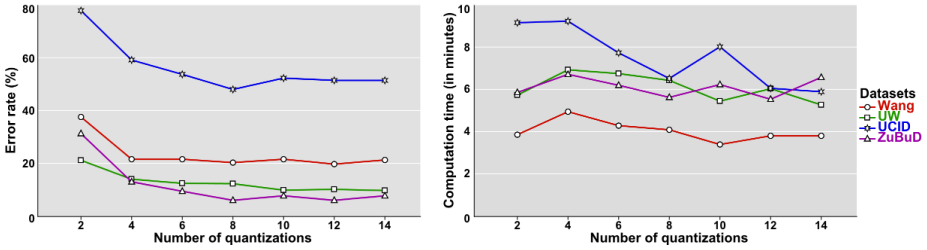

Fig. 6. Error rate (left) and computation time (right) of PixSearcher with different quantizations on the 4 benchmark datasets.

4.2 Selection of rescaled resolution

What is the best resolution to rescale original images? Figure 7 shows the test results of PixSearcher with different resolutions on the 4 image datasets. As depicted, in the left panel, we can get higher accuracy by increasing the resolution of rescaled images (increasing the details of quantized images). The improvements become insignificant with more than 40 pixels in each dimension (1). These improvements require the much higher computation time. The test results in the right panel indicate that our feature computation time grows quadratically with the rescaled image dimension (2). For these two reasons (1) and (2), we choose 40 by 40 resolution to rescale original images.

4.3 Comparisons of different CBIR algorithms

Figure 8 (left) shows the error rates for each algorithm for each image dataset. In particular, datasets are displayed vertically. For each dataset, we use dotplots to present the error rates. The lower error rates are closer to the left; the higher error rates are closer to the right. Red dots represent the error rates of PixSearcher; blue dots represent the error rates of color histogram; black dots represent the error rates of other algorithms. A recent survey [17] showed

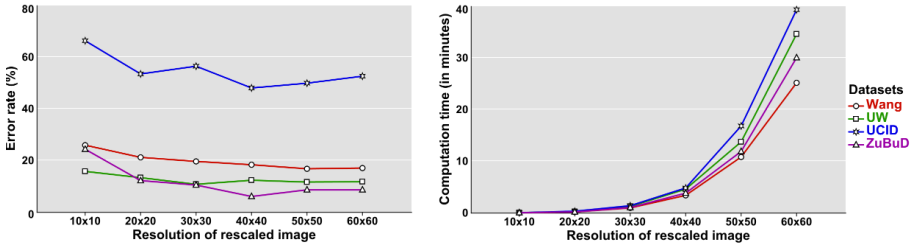

Fig. 7. Error rate (left) and computation time (right) of PixSearcher with different resolutions on the benchmark 4 datasets.

that color histogram outperforms other conventional algorithms. As depicted in Figure 8 (left), PixSearcher has similar error rate on UW dataset and lower error rates on UCID and ZuBuD datasets. On average of the 4 image datasets, PixSearcher has lower error rate (21.2%) than color histogram (22.1%).

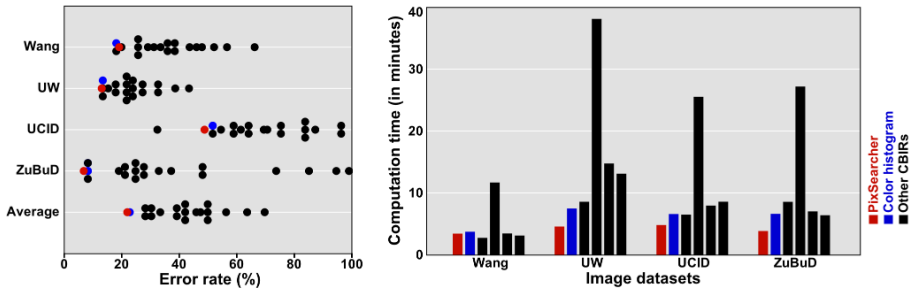

Fig. 8. Dotplots (left) display the error rates for 20 algorithms for 4 image datasets (Wang, UW, UCID, and ZuBud). Bar chart (right) compares computation time.

Figure 8 (right) shows the feature computation time for each algorithm. In particular, PixSearcher is in red, color histogram is in blue, other 4 algorithms (gray histogram, Tamura texture features, global texture features, and local features) are in black. PixSearch is shown to be the fastest algorithm in this group. These are the CBIR algorithms that we were able to configure to run on our machine. All tests were performed on a 2.3 GHz Intel Core i5, Mac OS X Version 10.7.5, 4 GB RAM running Java 1.6 and Processing 1.5.1.

5 Image Retrival examples

Figure 9 shows the retrieval results of PixSearcher versus other CBIR systems using the Corel database. This is a general-purpose image database including 60,000 pictures, which are stored in JPEG format with size 384 by 256. In particular, the query images are on the left. These images (one non-textured

and one textured) have been selected from the SIMPLIcity paper [8]. Due to the limitation of space, we show only the top 22 matches to each query on four rows associated to the four CBIR algorithms (PixSearcher, color histogram, Tamura texture features, and global features).

Fig. 9. Retrieval results of one non-textured image (top) and one textured image (bottom) in the Corel database.

6 Conclusions

In this paper, we proposed a set of visual features extended from scagnostics to handle pixels in images. The image descriptors are computed on pixels obtained by filtering the original images at different levels on Hue, Saturation, and Brightness. Our system, PixSearcher, outperformed conventional techniques on speed and accuracy (see Figure 8). Moreover, this model works even on black and white images (we use Brightness in this case), which the conventional color histogram can not.

For future work, we are planning to evaluate our approach on much larger image databases, for example the CoPhIR project [18]. The CoPhIR database contains 100 million images that have been specifically developed for testing on the scalability of image retrieval algorithms: http://cophir.isti.cnr.it/index.html.

References

1. Kato, T.: Database architecture for content-based image retrieval. In: Proceedings of SPIE 1662, Image Storage and Retrieval Systems. (1992) 112–123

2. Rubinstein, M., Liu, C., Freeman, W.T.: Annotation propagation in large image databases via dense image correspondence. In: Proceedings of the 12th European conference on Computer Vision - Volume Part III. ECCV'12, Berlin, Heidelberg, Springer-Verlag (2012) 85–99
3. Wilkinson, L., Anand, A., Grossman, R.: High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions. IEEE Transactions on Visualization and Computer Graphics **12** (2006) 1363–1372
4. 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. Computer **28** (1995) 23–32
5. Wang, J.Z., Li, J., Gray, R.M., Wiederhold, G.: Unsupervised multiresolution segmentation for images with low depth of field. IEEE Trans. Pattern Anal. Mach. Intell. **23** (2001) 85–90
6. Ma, W.Y., Manjunath, B.S.: Netra: a toolbox for navigating large image databases. In: Proceedings of the 1997 International Conference on Image Processing (ICIP '97) 3-Volume Set-Volume 1 - Volume 1. ICIP '97, Washington, DC, USA, IEEE Computer Society (1997) 568–
7. Carson, C., Thomas, M., Belongie, S., Hellerstein, J.M., Malik, J.: Blobworld: A system for region-based image indexing and retrieval. In: In Third International Conference on Visual Information Systems, Springer (1999) 509–516
8. Wang, J.Z., Li, J., Y, G.W.: Simplicity: Semantics-sensitive integrated matching for picture libraries. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 947–963
9. Weber, R., Mlivoncic, M.: Efficient region-based image retrieval. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management. CIKM '03, New York, NY, USA, ACM (2003) 69–76
10. Bartolini, I., Ciaccia, P., Patella, M.: A sound algorithm for region-based image retrieval using an index. In: Database and Expert Systems Applications, 2000. Proceedings. 11th International Workshop on. (2000) 930–934
11. Eidenberger, H.: How good are the visual mpeg-7 features? In: SPIE IEEE Visual Communications and Image Processing Conference. (2003) 476–488
12. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision-Volume 2. ICCV '99, Washington, DC, USA, IEEE Computer Society (1999) 1150–
13. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2. CVPR '06, Washington, DC, USA, IEEE Computer Society (2006) 2161–2168
14. Squire, D.M., Müller, W., Müller, H., Raki, J.: content-based query of image databases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback. In: Scandinavian Conference on Image Analysis. (1999) 143–149
15. Ware, C.: Visual Thinking For Design. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2008)
16. Hartigan, J.A., Mohanty, S.: The runt test for multimodality. Journal of Classification **9** (1992) 63–70
17. Deselaers, T., Keysers, D., Ney, H.: Features for image retrieval: An experimental comparison. Inf. Retr. **11** (2008) 77–107
18. Bolettieri, P., Esuli, A., Falchi, F., Lucchese, C., Perego, R., Piccioli, T., Rabitti, F.: CoPhIR: a test collection for content-based image retrieval. CoRR (2009)