A case for Image Querying through Image Spots

Peter Bosch, Arjen de Vries, Niels Nes, Martin Kersten

CWI, Netherlands

Abstract

We present image spot queries as an alternative for content-based image retrieval. Through image spots (*i.e.*, selective parts of images) users can better indicate which parts of the image are relevant for their query. Compared to traditional approaches, we allow users to search image databases for local image (spatial, color and color transition) characteristics rather than global features.

1 Introduction

In this paper we argue that image querying through image 'spots' is a better way of formulating an image query compared to traditional query formulation methods. We define an image spot as a 'representable' portion of an image and this spot is used to search alternatives in a (possibly) large image set. Any part of an image can be a spot, and henceforth there are many possible spots per image.

First generation image querying systems are based on color-based histogram matching. A sample image is characterized by its color contents and images are compared by, for example, computing the Euclidean distance similarity function between the images' color histograms. The fundamental problem with this approach is that often spatial considerations are ignored. Images that have the same color build-up as the sample image, but are semantically different are also presented as *good* answers.

Second generation content-based image retrieval approaches are those that also consider spatial constraints. A number of systems exist that combine both color and/or texture contents of images with spatial information and allow search operations on these 'features.' Unfortunately, it is usually computationally expensive to construct and search such a database. However, the general tendency is that when spatial constraints are used, the query results improve in quality.

By using image spots as our query points, we explicitly use the color components of salient objects in images as well as the spatial layout of the color components when we are searching through our database. Contrary to second generation systems we do not store objects in our indexes: instead we only store the spatial location of dominant colors of images. By carefully matching dominant colors and analyzing dominant-color transitions we reconstruct objects while performing query searches in the database.

Although the spot-based approach increases both the query and answer domain, we rely on the hypothesis that not all portions of an image are equally important for query formulation. Especially images containing many objects require more precision to articulate the query compared to traditional methods based on global image feature sets. Conversely, a multi-object query answer image may contain a small spot of high interest otherwise hidden behind the statistics of the complete image.

We are currently building up a database with eventually more than 1 M images to perform spot-search experiments. This project's aim is to build a system that is capable of steering users toward their 'correct' query through spot articulation. This implies that the system must be quick enough for interactive use, while at the same time it must be able to consider colors, color transitions and spatial layouts of the colors the image database.

2 Related work

An overview of QBIC [6, 5, 13], VisualSEEk [14], BlobWorld [3], color histogram refinement [10, 11], color correlograms [8], histograms with spatial information [16], Isis [7], and work on capacity and

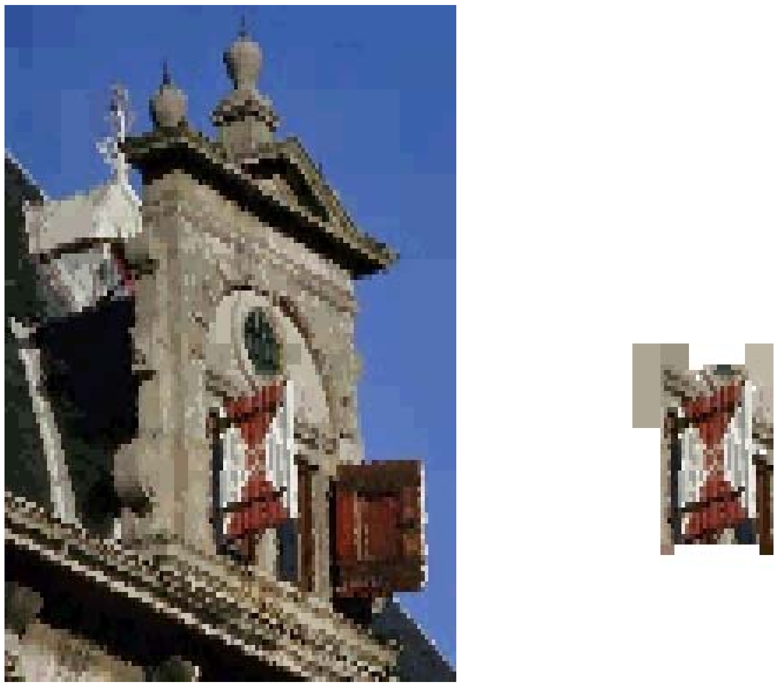

Figure 1: The original image and a single spot in quad-tree representation.

limits of high dimensional color spaces [17, 1].

3 Image spots

Before we can define the structure of spots, we first describe our image index since spots are defined as a subset of this index.

Our image database contains for each image a seperate dominant-color quad tree [12], much like as is done in Smith *et al.* [15] and by Lu *et al.* [9]. First, we convert a standard RGB image to a 256-bin HSV color space and we determine the dominant color of the overall image. When this color is available for at least 10% of the pixels in all four quadrants of the image, a color feature [1] is defined for that rectangle. Unlike previous approaches, we subsequently wipe out all pixels in the dominant color, and the algorithm proceeds with each of the four quadrants separately as a 'new' image. All image quad trees are stored in our main-memory database [2]. This database is large enough (64 Gb of main memory) to hold over 1 M images in main memory.

Formally we have defined a spot as a subset of an image quad tree. A spot may consist of a single spatially closed region or it may consist of multiple closed regions (a multi-spot). Figure 1 presents a simple spot example and the original image it was taken from. For simplicity all spots are multi-spots of one or more spatially disjoint image subsets.

Spots cannot be used directly for querying. When a spot is used for querying without normalizing the spatial locations, one can be sure that only the original image spot can be found: it is unlikely that there are other images that have exactly the same color contents at exactly the same position(s). Spots are normalized in two ways: the color contents of the HSV color space is reduced to a 16-bin Itten-Runge color space [4] and the spatial location is normalized.

We reduce the color space to a lower dimensionality to avoid using a too selective color space. As is also noted by Beyer *et al.* [1], using high dimensional indexes only leads to a query result of the original image – the remainder of the results has an Euclidean distance that is approximately equidistant to the

[1] A feature describes an image area and color of the area.

query point. Empirically we found that using the Itten-Runge space leads to results that we can visually understand.

The second spot transformation we perform is the spatial transformation. Given a spot, we determine the color transition strings in the spot; *i.e.* we determine which colors are co-located. The color transition strings are used as the actual query points. When there are multiple spatially disjoint areas in a spot, we record the relative position of the multiple areas to each-other in eight directions (North, North-East, etc.).

Our GUI *will be presented in the full version; it allows more extensive schemes to formulate the image spot queries.*

4 Query processing

Normalized spots are matched in our image database. We perform this search in three phases: pre-selection of images on color transitions, matching of spot geometrical centers lines and matching the entire spot contents.

The reason for performing searching in three steps is to reduce computational effort: the first phase can be performed quickly, the second operation takes more effort and the third operation is an expensive operation. It is important to split up these operations because we aim at interactive searching a database with over 1 M images. If it takes too long to provide an answer (*i.e.*, longer than 20–30 seconds), users will be quickly frustrated.

Our index is split into two parts: a primary index to search for color transitions and the image quad trees. The primary index is used to preselect images that satisfy particular 3-color transitions. Each 3-color transition in the Itten-Runge color space is implemented as a bit string and each image is assigned a bit. A bit is set when a particular color transition is present in the image. To test the availability of a particular spot color transition string in the images, bit strings are combined. Note that this combining leads to *false positives* since spatial consideration are ignored in this phase.

Once candidate images have been selected through the primary index, the geometrical center lines of each object within the multi-spot is matched to the quad trees. The algorithm first matches the geometrical center point of each object before it proceeds in all directions of the center lines. The algorithm performs this operation for each object in the multi-spot.

In the final step, the entire multi-spot is matched, and the results are ranked.

Details in the full version.

5 Experiments

We are testing and integrating each of the previously described components. We are populating our main-memory database with images and we are experimenting with spot querying. Currently our database holds approximately 60,000 images.

We have performed extensive analysis on our image index to learn of the nature of the images before devising a search algorithm. *Details in the full version.*

We have a rudimentary implementation of our system available and we are tuning our system to meet our performance goals. Preselecting images with the primary bit-vector index can be done in just a few hundred milliseconds for 1 M images. Matching the geometrical center lines onto a quad tree with 10,000 leafs is performed in 10–20 milliseconds, so it is imperative that pre-selection reduces the search space well. We are currently still implementing the full spot match.

Details in the full version.

6 Summary

In this paper we have presented spot queries as an alternative for content-based image retrieval. Our approach is that by using local image features, users are better in articulating which part of the image they are interested in. The selected 'spots' are used to search for alternatives in a three-step search algorithm in our main-memory database.

We have performed a small number of spot-based queries in our database. With these experiments we have learned how to perform the spot-based querying and we have analyzed the results we have obtained so far. These results stem hopeful: we can find similar spots to the query point.

References

[1] Kevin Beyer, Jonathan Goldstein, Rgahu Ramakrishnan, and Uri Shaft. When is 'Nearest Neighbor' Meaningful? *International Conference on Database Technology (ICDT) 1999*, pages 217–235. Springer-Verlag, 1999.

[2] Peter Bosch, Niels Nes, and Martin Kersten. Navigating through a forest of quad trees to spot images in a database. INS-R0007. Center for Mathematics and Computer Science (CWI), Amsterdam, 2000.

[3] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A System for Region-Based Image Indexing and Retrieval. *Third International Conference Visual Information and Information Systems*. Springer-Verlag, 1999.

[4] Alberto Del Bimbo. Chapter 2, Image retrieval by colour similarity. In *Visual Information Retrieval*, pages 97-99. Morgan Kaufmann Publishers, Inc., 1999.

[5] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and Effective Querying by Image Content. *Intelligent Information Systems*, **3**:231–62, 1994.

[6] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jon Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by Image and Video Content: the QBIC system. RJ-9949 (87908). IBM Research Division, Almaden Research Center, 30 March 1995.

[7] Theo Gevers and Arnold W.M. Smeulders. PicToSeek: Combining Color and Shape Invariant Features for Image Retrieval. *IEEE Transactions on Image Processing*, **9**(1):102–119. IEEE, January 2000.

[8] Jing Huang, S. Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zahib. Spatial Color Indexing and Applications. *International Journal of Computer Vision*, **35**(3):245–268, 1999.

[9] Hongjun Lu, Beng-Chin Ooi, and Kian-Lee Tan. Efficient Image Retrieval By Color Contents. *Applications of Databases* (Vadstena, Sweden), number 819 in Lecture Notes in Computer Science, pages 95–108. Springer-Verlag, June 1994.

[10] Greg Pass and Ramin Zahib. Histogram refinment for content-based image retrieval. *IEEE Workshop on Applications of Computer Vision*, pages 96–102. IEEE, 1996.

[11] Greg Pass and Ramin Zahib. Comparing Images Using Joint Histograms. *Journal of Multimedia Systems*, **7**(3):234–240, 1999.

[12] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison Wesley, 1990.

[13] Harpreet S. Sawhney and James L. Hafner. Efficient Color Histogram Indexing for Quadratic Form Distance Functions. RJ-9572 (83578). IBM Research Division, Almaden Research Center, 28 October 1993.

[14] John R. Smith and Shih-Fu Chang. Tools and Techniques for Color Image Retrieval. *SPIE*, volume 2670. SPIE, 1630-9.

[15] John R. Smith and Shih-Fu Chang. Quad-Tree Segmentation for Texture-Based Image Query. *ACM Multimedia 94* (San Francisco). ACM, October 1994.

[16] Markus Stricker and Alexander Dimai. Color Indexing with Weak Spatial Contraints. *SPIE'96*, pages 1–12, 1996.

[17] Markus Stricker and Michael Swain. The Capacity of Color Histogram Indexing. *CVPR'94*, pages 1–5, 1994.