Image Retrieval with Semantic Sketches

David Engel, Christian Herdtweck, Björn Browatzki and Cristóbal Curio

Max Planck Institute for Biological Cybernetics,
72076 Tübingen, Germany
{david.engel,christian.herdtweck,bjoern.browatzki,cristobal.curio}@tuebingen.mpg.de

Abstract. With increasingly large image databases, searching in them becomes an ever more difficult endeavor. Consequently, there is a need for advanced tools for image retrieval in a webscale context. Searching by tags becomes intractable in such scenarios as large numbers of images will correspond to queries such as "car and house and street". We present a novel approach that allows a user to search for images based on semantic sketches that describe the desired composition of the image. Our system operates on images with labels for a few high-level object categories, allowing us to search very fast with a minimal memory footprint. We employ a structure similar to random decision forests which avails a data-driven partitioning of the image space providing a search in logarithmic time with respect to the number of images. This makes our system applicable for large scale image search problems. We performed a user study that demonstrates the validity and usability of our approach.

Keywords: Content-Based Image Retrieval, Sketch Interface, Semantic Brushes, Real-Time Application, User-Study

1 Introduction

There are millions of image searches performed every day which to date rely primarily on text-based queries. With the advent of increasingly powerful computer vision systems for object detection, segmentation and tracking, and the introduction of large labeled image databases such as LabelMe [1], the opportunity arises for more advanced image retrieval tools to exploit this additional information. In this paper we introduce a novel image retrieval framework for finding images based on semantic sketches in large labeled databases.

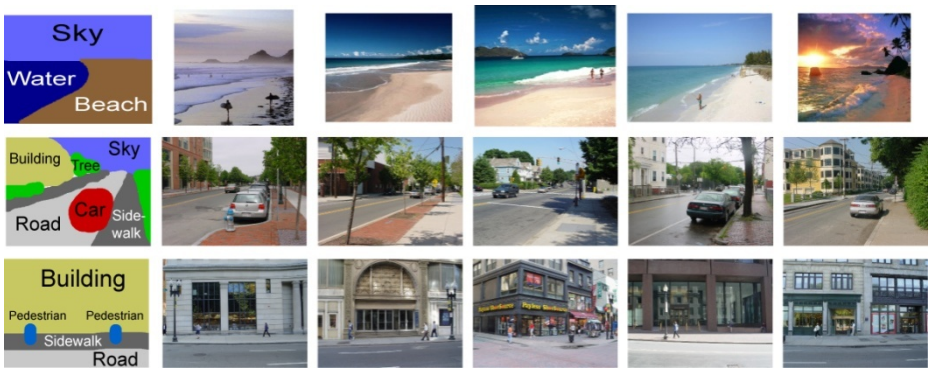

Fig. 1. Two example queries for street scene images and one for coastal images with their top five matches. Colors in the query sketch denote semantic classes. The text annotations are not part of the query and are shown here for illustration purposes only.

Traditionally, content-based image retrieval (CBIR) systems rely primarily on image statistics and machine learning techniques to select matching images from a database. This might not be the optimal way to approach the problem since it neglects sources of high-level information such as image annotations. Given the recent progress in computer vision it is reasonable to expect a steep rise in the number and availability of labeled images in the near future.

We propose a retrieval system that allows the user to formulate semantic queries intuitively rather than working with photometric queries. As opposed to many other sketch-based CBIR systems we do not require the user to draw detailed sketches of the objects. Our intuitive interface enables the user to indicate the semantic composition of the desired image with the help of semantic brushes (such as a brush for the classes “car” or “sky”). In such a scenario text-based searches (e.g., Google Image Search) would fail because they do not take into account the spatial relationships of the classes. Since we operate on high-level information, searches can be performed very efficiently using a tree structure, in contrast to linear methods which would be infeasible in large-scale retrieval scenarios. Our intended application is finding images that roughly match a user's wishes, not a target search for one specific image which would require higher developed sketching abilities from the user.

To evaluate and validate our system we performed a user-study with 10 participants. They were asked to sketch street scenes and rate the images that were retrieved by our system. For the user study and the algorithmic evaluations we used the StreetScenes database [2] which contains more than 3500 images with labels of eight classes (pedestrian, car, bicycle, street, sidewalk, building, sky and tree). This database provides a suitable testbed for our algorithm as it contains a large number of images from one scene category. It can be viewed as a dense sampling of a small part of image space akin to what would result if a computer vision algorithm would automatically label a vast quantity of images.

The main contributions of our method are: the usage of high-level semantic sketches, its computational efficiency which makes it applicable to large-scale image

searches, its robustness which leads to very low requirements on users sketching abilities, and its validity as demonstrated by a user study.

In Section 2 we describe previous work in this field and contrast the proposed framework to it. Section 3 describes the employed distance measures, the decision trees and the interface that comprises this system. Section 4 details the algorithmic evaluation and the user study that has been done to validate this method. Finally, Section 5 provides a summary and a discussion of future work.

2 Related Work

The need for systems that search images based on user queries has motivated a large body or research literature. In early studies the user was asked to specify the query in terms of visual features such as color or texture by drawing (query by image content, QBIC e.g. [3, 4]) or through example images (query by visual example, QBVE). The latter approach allows to calculate more complex image correlation measures (e.g. [5, 6]), yet, both approaches suffer from the so-called "semantic gap" i.e., the lack of correspondence between visual and semantic features. They yield results that have the desired low-level properties (e.g., containing a black shape) but may not fulfill the user's semantic wishes (e.g., containing a black dog versus containing a black car). For a concise review of earlier CBIR approaches and an extensive list of references, the reader is referred to [7].

One approach to bridge this "semantic gap" is to use text-based queries as offered by image search services such as *Google*, *Bing* or *Flickr*. These approaches employ semantics in form of keywords that are assigned to images, text surrounding images in web pages, as well as manually and automatically created annotations of objects, regions and scene classes. Purely text-based systems, however, do not allow the user to specify an image composition. This can be addressed by allowing the user to draw a query image using regions of photometric patches (e.g., [8]). These patches are generated in an unsupervised fashion from training data. This is a recent approach of a query by semantic example (QBSE) technique. In systems such as [9] the user specifies an image from which a computer vision system extracts semantic properties. These properties are then compared to the images in the database to retrieve images that are semantically similar to the query image. Such systems are often not applicable for real-time searches in large databases, because of the difficult similarity judgments needed to find matches. Further reviews on semantic image retrieval can be found in [10, 11].

A quite different approach to the problem of image retrieval is photo montage or photo synthesis which aims to create the image the user has in mind instead of searching for a similar image in a database. The systems described in [12] and [13] allow the user to specify semantic regions similar to our system. Based on this input the system automatically retrieves image parts and stitches them together to form a coherent image. Sketch2Photo [14] also lets the users specify objects at any position in the image, but also requires them to sketch the objects themselves and annotate them with text labels. This gives the user the freedom to use any object label that an internet image search can reasonably retrieve images for, and also allows finer control

over the objects' appearance, but requires good sketching abilities and iterative refinement of the results.

Finally, the usability of image retrieval tools has been the topic of research. A review on semantic search tools can be found in [15], for references on image retrieval systems and their evaluation confer to [16, 17].

3 Methods

Our retrieval algorithm employs a tree structure similar to a random decision forest [18] to retrieve candidate matches from the database and a fine grained search through the returned matches to determine a ranking. This scheme allows it to perform very fast searches (on average less than one millisecond per search in a tree containing one million images on a current office PC) with a complexity of $O(m \log n)$ where m is the number of trees and n the number of images (cf. Fig. 7).

3.1 Matching-cost function

Central to our image retrieval system is the definition of a cost function $C(Q, I)$ which measures the quality of the match between a query sketch Q and an image I with annotation A. This cost function allows the algorithm to rank the images and present a few top-ranking search results to the user. Desirable features of the cost function are a high correlation between the returned matches and the image the user had in mind (which will be discussed in Section 4.2), and robustness against the bad drawing skills of the average user (see Section 4.1 and Fig. 8).

We define a straight-forward cost function that can be evaluated very quickly using the scalar product between query sketch and distance transformation summed up over all classes in the image and the sketch: $C(Q, I) = \sum_i^N \langle Q_i, D_i \rangle$. An intuitive interpretation of this scheme is that we accumulate the distance that each pixel in the query image Q_i has to travel to the nearest pixel containing the respective object in the image. Using the distance transformation to calculate the cost function affords the nice properties of being intuitively plausible and making the search robust against imprecise sketching. If the annotation A of the image does not contain all objects that are present in the query Q, we add a high penalty κ to the cost function:

$$C(Q, I) = \sum_i^N \langle Q_i, D_i \rangle + \kappa(Q, A)$$

$$\kappa(Q, A) = \begin{cases} 0, & if\ all\ classes\ from\ Q\ exist\ in\ A \\ const, & otherwise \end{cases}$$

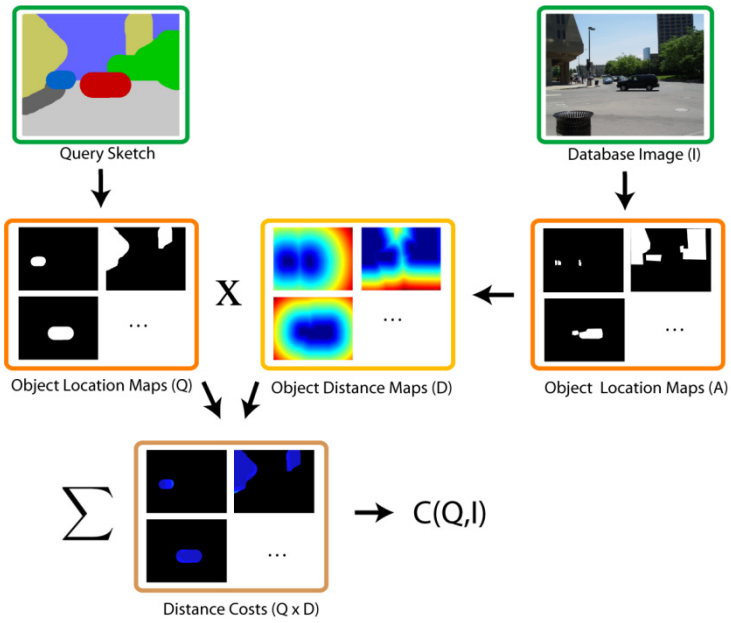

Fig. 2 Pipeline for evaluating the matching cost between an image $\boldsymbol{I}$ and a query sketch $\boldsymbol{Q}$: for each semantic class a binary object location map is created. A distance transformation is applied. It yields a map that contains the distance from each pixel to the nearest pixel of an object. The query sketch is also translated into a set of object location maps which are multiplied pixelwise with the distance maps. Summing over all image locations and classes results in the matching cost.

A depiction of the evaluation of the cost function is shown in Fig. 2. Using such an intuitively plausible cost function allows us to easily optimize and augment our system. Linear weighting of the different object classes and adding additional cost terms to represent possible further query properties such as the color of the objects are straightforward augmentations of the cost function. Fig. 3 shows how the results are reweighted when adding a cost term for color $C_{col}(Q_{col}, I)$. We implemented this cost term by computing the earth mover distance (EMD) between the color histogram Q_{col} the user has specified for each region (black regions denote regions where the user does not care about the color) and the normalized histogram of the region in the image where the object is present. The EMD provides a suitable distance measure between two color histograms and yields better results than for example comparing the mean colors of two regions:

$$C_{col}(Q_{col}, I) = \sum_{i}^{N} EMD\left(Q_{col,i}, hist(A_i \times I)\right) .$$

The augmented cost function is as follows:

$$C(Q, Q_{col}, I) = \sum_{i}^{N} \alpha_i \times \langle Q_i, D_i \rangle + C_{col}(Q_{col}, I) + \kappa(Q, A) .$$

Fig. 3. Constraining the image search: The top row shows a search for a car on a street in front of a building. The second row shows the results when searching for the same configuration with the additional constraint that the car should be white. The third row shows the same search, but constrained to gray buildings.

At this point the α's are set by hand. As the main evaluation criterion for such a system is how well the results match what the user had in mind they could easily be optimized with further user studies.

3.2. Decision trees

For small datasets that contain only a few thousand images the evaluation of the cost function for each image is feasible albeit time-consuming. As the size of the labeled dataset grows, a linear search becomes intractable especially in the context of webscale applications. However, for most such applications we do not need the full ranking of all the images in the database but only need to retrieve a couple of the top ranked images. Since the similarity of two images depends critically on the query (e.g. which image properties we are looking for) we cannot examine the similarities for each class separately but have to take the whole image into account at each decision node. This observation implies that an algorithm cannot factorize the problem, which makes it exponential in its nature.

We address the problem of retrieving a couple of high ranking matches by using a heuristic inspired by random decision forests [18] (see Figure 4). Each random decision tree in our forest contains a pivot annotation $a \in A$ and a threshold σ. At this node the cost function $C(Q, I)$ for an incoming element is evaluated. If it is smaller than σ it is forwarded to the left child and otherwise to the right child of the node. During creation of the trees we select a random pivot element, calculate the costs to all images at that node and take the median of the costs as threshold σ. Taking the median guarantees that the resulting tree will be balanced and all searches can be performed in $O(\log n)$ time. We stop splitting the nodes when the number of elements at one node drops below a threshold (in our case five images). Consequently, any search in a tree returns one to four candidate matches, but further matches can be retrieved by traversing the tree backwards (backtracking).

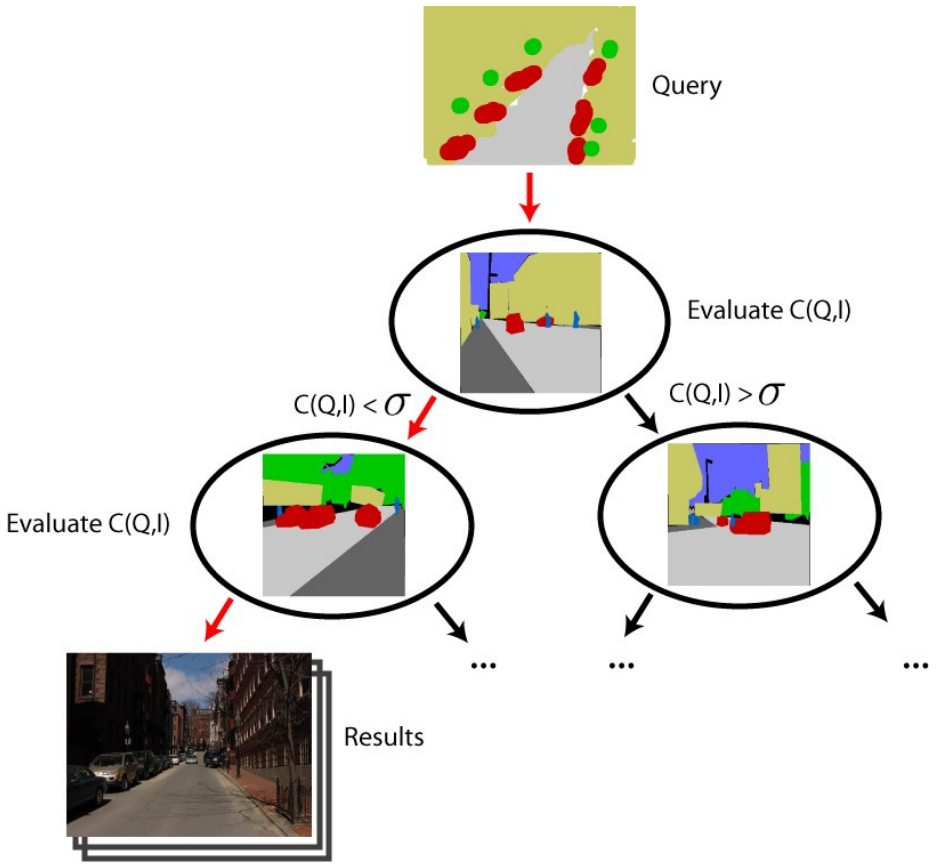

Fig. 4. Illustration of our decision tree. At each node the matching cost to a given exemplar is computed. The threshold $\boldsymbol{\sigma}$ determines whether the left or right child is visited next.

By evaluating the cost function with respect to a pivot element at each node we achieve a data-driven partitioning of the image annotation space. Different pivot elements lead to different trees which highlight different aspects of the cost function. To make full use of this we search through a forest containing m trees (usually 20). We obtain the final presentation order by ranking all returned results according to the cost function. Consequently, the total runtime of a search is $O(m \times \log n + m \times k)$ where k is the time it takes to evaluate the cost function for the resulting matches returned by one tree.

We show that the results returned by our forest approximate the matches returned by a linear search in Section 4.1. We further demonstrate that our distance measure is highly correlated with subject ratings of the user study described in Section 4.2. Together, these results show that our approach is a valid scheme to quickly retrieve images from a database based on semantic sketches.

3.3 The Sketch Search interface

We created a painting tool that enables the user to specify the composition of desired images (cf. Fig. 5). The tool offers a collection of semantic brushes that allow the user to specify object locations. These brushes can represent objects such as cars or bicycles but also image regions such as sky or road. To distinguish different brushes we assign a unique color to each of them. The color value itself is not relevant, since color is only used to denote the type and location of an object, not its appearance. Which brushes are available depends on the labels found in the current image database. For our evaluations we used the eight classes contained in the StreetScenes database (see Section 4).

Fig. 5. Sketch Search user interface. The user chooses objects to include in the scene from the object palette on the left. The canvas depicts a drawn street scene that is used as query for the image search. It shows a street scene containing a road, a sidewalk, two cars, two pedestrians, buildings, trees and sky.

To create a query image the user selects scene elements and draws them onto a canvas. The handling of the tool resembles that of common image editing programs. This way of composing a scene is intuitive and self-explanatory, as confirmed by the participants in our user study (see Section 4.2).

To specify an object it is not necessary to draw its precise shape. It is sufficient to mark the region in the image where instances of its class should occur. By drawing two cars, as shown in the example sketch in Fig. 5, the user specifies that the two

areas marked as “car” (red) should contain objects labeled as “car”. This constraint is fulfilled as long as there is at least one matching object in each of those regions, thus allowing the presence of cars in other regions.

4 Evaluation

In this Section we present the evaluation of our system. We evaluate and validate our method using the StreetScenes database [2] which contains 3547 images taken in Boston together with annotations of eight major classes: Pedestrian, Bicycle, Car, Street, Sidewalk, Building (including stores), Sky and Trees. Note that our algorithm does not distinguish objects (e.g. cars, people) and semantic regions (e.g. sky, street) and deals with them quite naturally. These classes are well suited for our envisioned application since they could potentially be labeled automatically by a computer vision algorithm. The number of categories might seem small when compared with the much higher number of classes found for example in the LabelMe database. But, most of the classes from LabelMe are irrelevant for the proposed large-scale image retrieval task as they have too few occurrences in the database.

Obtaining human annotations is expensive and time consuming which implies that our algorithm would mainly operate on computer generated labelings. These annotations would only contain labels for categories that are accessible to vision algorithms, which amounts to a few high-level categories for the near future. Gender specific searches are for example not plausible since algorithmic differentiation between male and female persons is a difficult task. Lastly, it has to be mentioned that there is a strong correlation between classes naturally occurring in images (e.g. “cars” and “road” often appear together in an image while “car” and “table" do not). This further reduces the number of classes that have to be present in a tree.

4.1 Algorithmic evaluation

As first evaluation we perform automated searches using queries formed from ground truth annotations from the database. We plotted the results produced by our algorithm and the results obtained when evaluating the cost function for each image individually in Fig. 6. The results show that our algorithm is able to approximate the linear search through the database with a logarithmic search in our random decision forest.

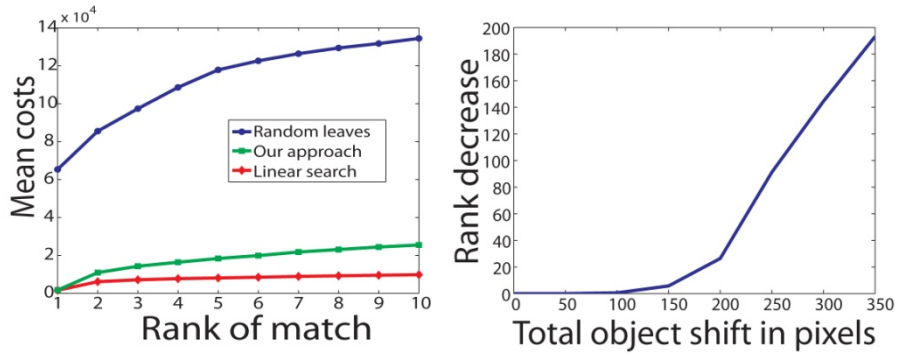

Fig. 6. Left: Comparison between results for our approach, a linear search through all images and a ranking of random tree leaves according to our cost function. The graph shows the mean costs associated with the top 10 matches. **Right**: The figure shows the decrease in retrieval performance for shifts of object location. The average decrease of matching rank compared to the unshifted queries is plotted on the y-axis.

Insensitivity to variation in object positions in query images is crucial to provide a level of robustness that is needed to deal with the low fidelity of user sketches (see Section 4.3). We evaluated the robustness of our system by shifting the positions of object classes in horizontal or vertical direction by a random amount. However, we set the total number of pixels that all object classes were shifted in one image to a fixed value. In this case a shift by 320 pixels means that each of the eight object classes we use is shifted by 40 pixels on average. Fig. 6 shows the decrease in retrieval performance with respect to the original queries. For smaller shifts up to approximately 150 pixels the results differ only slightly from those obtained by the original queries. It is important to note that query images had a dimension of 320x320 pixels. Shifting an object class by a large distance can result in its complete removal from the scene, contributing to the steeper performance decrease at summed shifts of more than 200 pixels (which can be expected to be intended by the user). The evaluation shows that our algorithm is robust against variations in the sketched object location up to a certain degree.

For eight classes and a 32x32 descriptor each image can be represented using 8096 Byte. This means that a million images would take up 8GB of memory. This memory problem is present for every image retrieval system as they need to keep the image descriptors in memory during runtime. Fortunately, due to the linear nature of the distance transformation the dimensionality can be greatly reduced using a principle component analysis (PCA). Using PCA we can encode 99% of the variance of the descriptors with just 34 dimensions for this highly redundant data-set. When stored in a single precision float vector the compressed descriptor takes up only 136 Byte. The evaluation of the cost function is then preceded by a matrix multiplication and addition of the means to project the principle components scores of the descriptors back into the image space. This has to be done only log n times for each tree, thus, not influencing the runtime critically. This effectively removes the memory problem, making our algorithm applicable for large scale image search problems.

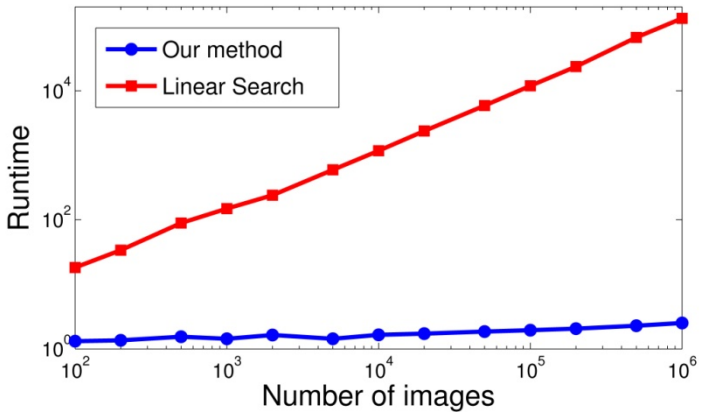

Fig. 7. Comparison between searches using our method and a linear search in databases of different sizes. Graph axes are logarithmic.

To evaluate the speed of our search we built trees for a more realistic search setting. By mirroring, shifting and subwindowing random images from the StreetScenes database we created a dataset containing one million images. All timing studies were conducted on a regular office PC with a 3GHz dual core processor and 2GB of RAM using our MATLAB™ implementation of the search algorithm. Creating a tree for this dataset takes on average 3 minutes. Queries in this tree take on average 0.23 milliseconds. A linear search through all image descriptors on the other hand takes 17.4 seconds on average making it unsuitable for any large scale application. For more details on computational efficiency see Fig. 7.

4.2 User study

In this section we show that our approach retrieves images that are not only good in an algorithmic sense but also in a subjective sense which is of key importance for a retrieval system. We conducted a user study with ten participants (7 female, 3 male, mean age was 25.6). On average each participant did 45.7 trials in 60 minutes. The drawing phase took 53 seconds on average. To ensure that participants would sketch different images in each trial they were shown an "inspiration" image taken randomly from the StreetScenes database for one second before the drawing phase started. The subjects were explicitly instructed to take this image solely as an inspiration and not to search for this image. In the drawing phase users then sketched an image using the tool described in Section 3.3. The resulting sketch served as input for our algorithm which returned images using a forest consisting of 20 trees.

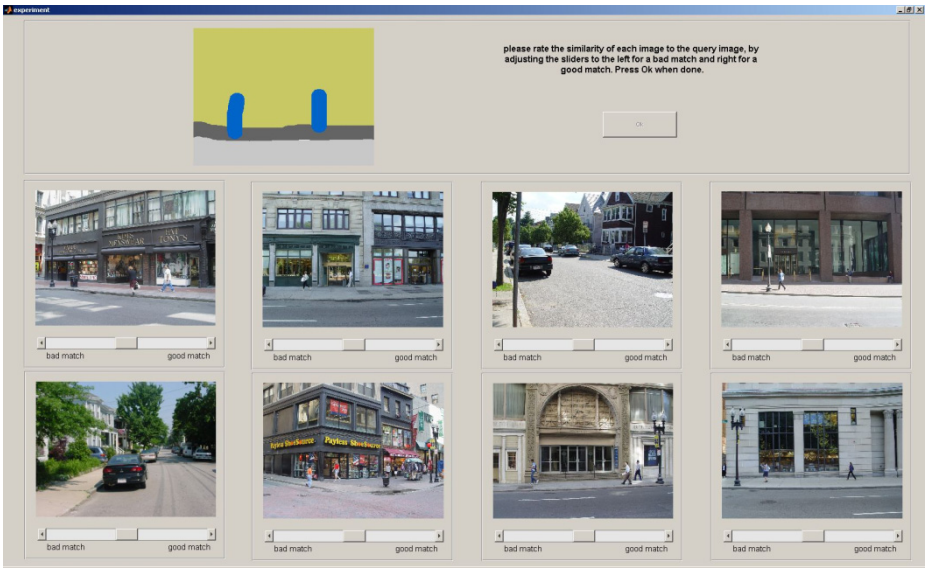

Fig. 8. Example query and resulting images in the GUI from our user study. The third image in the first row and the first image in the second are random images, our algorithm's best matches are the second image in the first row and the last in the second row.

For the evaluation phase we selected the four best matches retrieved by our algorithm and the worst and median matches retrieved by our algorithm as well as two random images from the database. We presented the images simultaneously at randomized positions in the GUI shown in Fig. 8. Note that "worst" and "median" here does not refer to the worst/median match in the whole database but just in the subset (on average 54 images) of potential matches returned by the search. Participants were then asked to rate the similarity of the images to their sketch on a scale of 1 (bad match) to 7 (good match). Participants were explicitly instructed not to rate the similarity to the original "inspirational" image which might have created confounds when users remember only certain details about the images.

For data analysis, ratings were normalized to a mean of zero and standard deviation of one. We then calculated the average participant rating for each of the seven classes of shown images (best match, second/third/fourth best match, mean match, worst match, random image). The results are visualized in Fig. 9. A post-hoc Scheffé test (significance level $\alpha = 1\%$) between the means confirms that participants gave significantly higher ratings to images that our algorithm considered to be good matches than to random images or bad matches. Furthermore, the average rating for the best match is significantly better than the rest while the second, third and fourth matches show no significant differences. The mean rating for a "median" match is significantly worse than the top-matches and significantly better than the worst and random matches. This is to be expected as the worst of the retrieved images often did not contain all classes from the sketch.

We further analyzed directly the relation between the algorithm's cost function and the participants' ratings. In Fig. 9 we visualize the mean costs over all query results compared to the average participant ratings for them. The y-axis of the plot is normalized to one standard deviation of all participants' responses. There is a strong correlation, which is partly due to the cost term κ which penalizes the absence of objects in bad matches.

In summary, these results confirm that our algorithmic definition of a good match coincides with the human intuition, allowing our system to yield user-intended results.

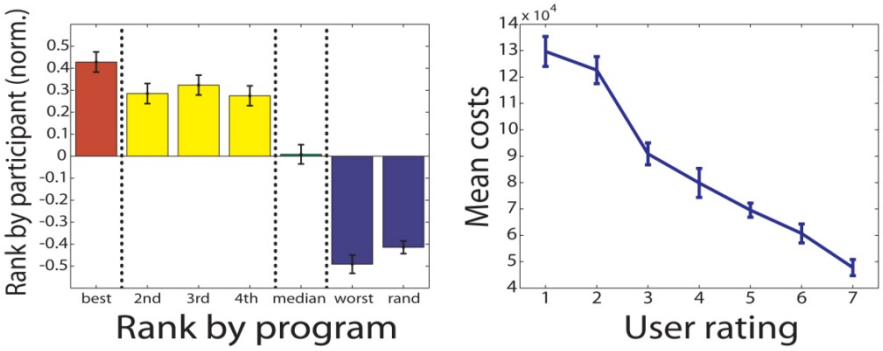

Fig. 9. Left: Normalized ratings of participants for our algorithm's top four matches, the "median" and "worst" match (see text) and two random images. Colors indicate a grouping of bars: bar heights in different groups are significantly different while bar heights in the same group are not. **Right:** The average costs that our algorithm assigns to potential matches show a strong correlation with participant ratings for these matches (1=bad, 7=good match). In both plots errorbars denote standard error.

4.3 Cognitive constraints

Our user-study showed that the manual generative abilities of participants are considerably worse than their visual discriminative abilities. Fig. 10 shows some examples of images and corresponding sketches users have drawn during the study. As people are able to copy an image shown during the sketching phase, the problem seems to occur when users have to create a 2D representation of a 3D scene they have in mind. Similar observations on participants failing to reproduce the correct perspective of a scene from memory have also been shown in psychophysical studies such as [19]. This poses a general problem to most systems that rely on sketch based interfaces. Such systems are expected to produce an image matching the one the user has in mind based solely on a potentially inaccurate sketch. Our system addresses this problem by using only rough semantic sketches and a distance transformation, both of which help to suppress errors users make during sketching.

Fig. 10. Failure case: "inspirational" image at the top left and sketch created from it in the user study on the bottom left; on the right the four best matches. For some of the bad user sketches (e.g., with impossible perspective as here), no reasonable matches can be found in the database, resulting in relatively bad results.

We feel that our interface that requires only rough sketches is already at the upper bound of what can be expected from average users, especially when looking at the errors in perspective that the users make during drawing. More complex sketching systems for purposes such as "Sketch2Photo" [13] allow users to create visually pleasing images with arbitrary scene compositions. However, they only work if the user is able to produce a sensible perspective composition otherwise they can only create an unrealistic image. Our system cannot generate images de-novo but is able to find sensible matches very efficiently and robustly which seems to be a better approach.

5 Conclusions

In this paper we have presented a novel system for content-based image retrieval using semantic sketches. By utilizing a fast straight-forward cost function together with a decision tree with guaranteed depth of $O(\log n)$ we are able to provide a very fast tool. Our system yields comparable results to a linear search thus being applicable for large scale image databases. In our user-study, we have demonstrated that our system and the cost function is usable and return images matching what the user drew. These properties demonstrate that our system is useful for image retrieval in large labeled databases.

In the future we plan to do a larger scale user study in order to optimize the parameters of the cost function and accommodate user sketching behavior. Furthermore, we will apply this retrieval system in tandem with a computer vision system that provides a coarse labeling in order to provide a sketch based search tool that operates on a database of millions of images.

6 References

[1] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation", *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, 2008.

[2] S. M. Bileschi, "Streetscenes: towards scene understanding in still images", Ph.D. dissertation, Massachusetts Institute of Technology, 2006.

[3] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image and video content: The qbic system", *Computer*, vol. 28, no. 9, pp. 23–32, 1995.

[4] C. E. Jacobs, A. Finkelstein, and D. H. Salesin, "Fast multiresolution image querying", in *Computer Graphics and Interactive Techniques*, 1995, pp. 277–286.

[5] R. Marée, P. Geurts, and L. Wehenkel, "Content-based image retrieval by indexing random subwindows with randomized trees", in *Proceedings of the Asian Conference on Computer Vision*. Springer, 2007, pp. 611–620.

[6] K. Hirata and T. Kato, "Query by visual example - content based image retrieval", in *Proc. Intl. Conference on Extending Database Technology*. Springer, 1992, pp. 56–71.

[7] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years", *Pattern Analysis and Machine Intelligence*, pp. 1349–1380, 2000.

[8] J. Fauqueur and N. Boujemaa, "Logical query composition from local visual feature thesaurus", in *Content-Based Multimedia Indexing*, 2003.

[9] N. Rasiwasia, P. L. Moreno, and N. Vasconcelos, "Bridging the gap: Query by semantic example", *IEEE Transactions on Multimedia*, vol. 9, no. 5, pp. 923–938, 2007.

[10] R. Datta, J. Li, and J. Z. Wang, "Content-based image retrieval: approaches and trends of the new age", in *Workshop on Multimedia Information Retrieval*, 2005, pp. 253–262.

[11] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics", *Pattern Recognition*, vol. 40, no. 1, pp. 262–282, 2007.

[12] M. Johnson, G. Brostow, J. Shotton, O. Arandjelovic, V. Kwatra, and R. Cipolla, "Semantic photo synthesis", *Computer Graphics Forum*, vol. 25, no. 3, pp. 407–413, 2006.

[13] N. Diakopoulos, I. Essa, and R. Jain, "Content based image synthesis", in *International Conference on Image and Video Retrieval*, 2004, pp. 299–307.

[14] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu, "Sketch2photo: Internet image montage", in *ACM Transactions on Graphics*, vol. 28, no. 5. ACM, 2009, pp. 1–10.

[15] V. Uren, Y. Lei, V. Lopez, H. Liu, E. Motta, and M. Giordano, "The usability of semantic search tools: a review", *The Knowledge Engineering Review*, vol. 22, no. 04, pp. 361–377, 2007.

[16] N. V. Shirahatti and K. Barnard, "Evaluating image retrieval", in *Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 955–961.

[17] J. Vogel and B. Schiele, “On performance characterization and optimization for image retrieval”, in *European Conference on Computer Vision*. Springer, 2002, pp. 49–66.

[18] T. K. Ho, “Random decision forests”, in *Conference on Document Analysis and Recognition*, 1995, p. 278.

[19] F. H. Previc and H. Intraub, “Vertical biases in scene memory”, *Neuropsychologia*, vol. 35, no. 12, pp. 1513–1517, 1997.