Experiences with QbS: Challenges and Evaluation of Known Image Search based on User-Drawn Sketches*

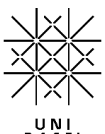
Michael Springmann Ihab Al Kabary Heiko Schuldt

Technical Report CS-2010-001
University of Basel
Email: {michael.springmann|ihab.alkabary|heiko.schuldt}@unibas.ch

Abstract

With the increasingly growing size of digital image collections, known image search is gaining more and more importance. Especially in collections where individual objects are not tagged with metadata describing their content, content-based image retrieval (CBIR) is a promising approach. However, the application of CBIR to known item search usually suffers from the unavailability of query images that are good enough to express the user's information need. In this technical report, we present the QbS system that provides content-based search in large image collections based on user-drawn sketches. The QbS system combines angular radial partitioning for the extraction of features in the user-provided sketch, taking into account the spatial distribution of edges, and the image distortion model. This combination offers several highly relevant invariances that allow the query sketch to slightly deviate from the searched image in terms of rotation, translation, relative size, and/or unknown objects in the background. To illustrate the benefits of the QbS approach, we present search results from the evaluation of our system on the basis of the MIRFLICKR collection with 25,000 objects and compare the retrieval results of pure metadata-driven approaches, pure content-based retrieval using different sketches, and combinations thereof.

Keywords:
Content-based image retrieval, query by sketch, known item search.



*The work has been supported by the Swiss National Science Foundation (SNF) in the context of the project *PAD-IR* under contract No. 200020_126829 / 1.

Computer Science Department
Bernoullistrasse 16, CH - 4056 Basel Switzerland.
Databases & Information Systems Group
Website: http://dbis.cs.unibas.ch

1 Introduction

Searching for information can be categorized in two fundamentally different classes with respect to the knowledge a user has about the information item(s) and also with respect to the conditions under which the search task can end successfully.

Searching for known items: a user knows that the items exist and has probably seen them before. The search task will end successfully only if the user has found all these items.

Searching for (potentially) novel items: a user is looking for items that satisfy her information need. The search task ends successfully as soon as the user is provided with some (not necessarily all) items that reflect the information she is looking for.

Depending on the *interaction intentions* [36] associated with these classes, different information seeking strategies have to be applied. The probably most effective strategy for finding a known image in an image collection is to remember an access path to it. A user might recall the folder in which the digital image is stored as a file and under which name, or she might have memorized how she accessed the image before (e.g., in case of a web search, which term led to the page containing the image). This of course only works for static collections in which the access path is kept stable over time. Yet exactly the same strategy, based on known access paths, will obviously not be applicable for novel item search. In order to achieve satisfactory results, the user would need a *shift in information seeking strategies* [36] e.g., by means of browsing the image collection to find novel items.

The strategy of browsing can of course also be applied for known item search, but it is certainly not as effective and may also not lead to a successful result if it is not known in advance whether the collection contains the searched item or not. In such a situation, the user may perform a *shift in interaction intention* [36], for instance, if the user does not necessarily need the known item, she may search for similar items instead.

For finding both, known images and novel images, content-based image retrieval (CBIR) provides powerful tools. However, CBIR requires a query image to start with that is sufficiently close to the final result, i.e., that precisely expresses the user's information need. Without such a query image, it is difficult or nearly impossible to achieve good retrieval quality, even if sophisticated relevance feedback mechanisms are available. *Query by Sketching* [13, 11, 7] addresses this problem and uses user generated sketches as query images. So far, two main problems have significantly impacted the successful application of query by sketching to known item search and novel item search. First, the mouse as most widely available input device limits the user-friendliness and expressiveness for drawing sketches. Second, users usually do not sketch complete images but concentrate on the parts which are most interesting for them. Thus, when comparing user-drawn sketches with images, there will usually be parts of the images to be queried that do not have any corresponding part in the sketch as the user may not remember or not be able to draw all details of image in the sketch. The user may also place the sketch not exactly at the right position, with proper scale, and/or orientation. Therefore, the corresponding parts may not be at the same coordinates in sketch and image. In order to successfully apply query by sketching to CBIR, both problems need to be solved jointly.

In this technical report, we present the QbS approach to query by sketching that exploits novel user interface technologies and supports various invariances in the comparison of a user-generated sketch and the images to make the query process robust against trans-

lations, rotations, and different scales. We have applied the results of our work to use cases from known image search and provide detailed evaluation results of the QbS system based on the MIRFLICKR-25000 image collection [14]. We intentionally focus on the application of query by sketching for known image search as this allows us to concentrate on particular interaction intentions of the user. This also addresses some of the very critical remarks on previous attempts to CBIR and their evaluation in [24].

The report is organized as follows: Section 2 surveys related approaches to query by sketching. Section 3 summarizes the challenges that have to be met when searching for known images based on sketches. In Section 4, we introduce in detail the QbS approach and explain how the challenges are addressed. Section 5 describes the implementation of the QbS system and Section 6 the results of the experimental evaluation. Section 7 concludes. Appendices A-C present the MIRFLICKR images that have been chosen for the evaluation of QbS, the sketches that have been produced, and their edge map representations.

2 Related Work

As the absence of a good example to start a query has been a problem from the very start of CBIR, query by sketching has always been of interest in this domain. However, it has not been as successful as other approaches to CBIR, mainly due to the unavailability of appropriate input devices. For the comparison of images and sketches, several approaches have been proposed.

The QVE system [13] uses a reduced resolution of the image and edge detection to generate the so-called abstract images in a size of 64×64 pixels. These are compared to a rough sketch provided by the user by aligning blocks of 8×8 pixels within a range to provide limited invariance to translation. The score for ranking the results is determined by computing the overall correlation between all blocks.

The QBIC system [21, 11] extracts several features from the images in the database including a global 256-bin color histogram, 20 shape features of manually or semi-automatically identified objects like area, circularity, eccentricity, major axis orientation and algebraic moment invariants, as well as global texture like coarseness, contrast, and directionality. The user can either specify (even partial) color distribution or draw a binary silhouette image of the shape using a polygon drawing routine.

Multiresolution wavelet decompositions of the color channels of sketches and images have been proposed in [16]. It is used for the desktop applications imgSeek[1] and digiKam[2], and the online application retrievr[3].

Elastic matching of the sketch to the edges detected in the image in [7] is performed by interpreting them as B-splines with 20 knots, not individual strokes or pixels. The approach is able to cope with differences in scale and attempts to deal with small rotations (of the order of 12 to 15 degrees). It can also be used to find several objects in a spatial relationship. To reduce computational complexity, filtering based on the aspect ratio of the object and the spatial relationship between objects is applied if more than one object is used. In [1], this approach is extended to use tokens (so that parts of an object can be

[1] http://www.imgseek.net/
[2] http://www.digikam.org/
[3] http://labs.systemone.at/retrievr/

retrieved) and a modified M-tree index structure is proposed for improved retrieval times.

Edgelet features are used for the detection of objects in [35] and humans in [34]. An edgelet is a short line or curve segment that can be quantized in a few edges describing categories. It is well-suited for finding similar curves inside an image but requires training to adapt to a particular problem domain as it applies machine learning methods to learn strong classifiers.

Various systems use the features and distance measures defined in the MPEG-7 standard, as for instance described in [23, 28]. The standard defines a number of descriptors that can be used for various applications, some of them better suited for the use in query by sketch than others. Color information, in particular Color Layout Descriptor (CLD) was used for sketches in [33], but results have not been satisfactory. For the use of shapes with closed contours, Curvature Scale Space (CSS) descriptors can be used. For plain edge information, the Edge Histogram Descriptor (EHD) based on edge point distribution according to image block partitioning and the amount of dominant edge directions in these partitions can be used. EHD considers five different edge directions, and it does not provide rotation invariance. Angular Radial Transform (ART) provides a 36-dimensional feature vector that consists of coefficients of sinusoidal base functions of the image and provides rotation invariance.

Angular Radial Partitioning (ARP) proposed in [3] has shown better results in experiments for sketch-based image retrieval than EHD and ART, while being also fast in extraction and search. The approach is based on pure spatial distribution of edge pixels in the so-called edge maps (or pictoral index). Such edge maps are generated from images which have been normalized to a common size and processed with a variant of the Canny edge detector on the luminance channel with a single threshold value β, to control the level of detail which will be preserved. Sketches are processed with skinning and noise reduction operators before comparing the sketches with the egde maps with regard to the number of edge pixels in each of the angular radial partitions. Rotation invariance can be achieved by applying a 1D FFT on the ARP feature vector, distance is computed using the Manhattan distance.

A different approach, much closer to the approach used in QVE, has originally been proposed for the recognition of handwritten characters: The image distortion model [18, 17] uses a reduced resolution of the image, commonly scaled to 32 pixels in height, while preserving the image's aspect ratio. Image deformations are allowed within the so-called warp range for individual pixels and the area around them (the so-called local context). Usually, the warp range is set to two, resulting in a 5×5 area allowed for deformation and a local context of one, resulting in a 3×3 area over which Euclidean distance is averaged and provides a sound graphical interpretation of distance as deformation. It is either applied on the grey intensities of the pixels directly or on the edges identified with a horizontal and a vertical Sobel filter. Compared to QVE, it uses a sliding window instead of dividing the image into blocks and by default a much smaller resolution. This approach can also be used to compare sketches with images by applying edge detection to the images.

More recently, a novel feature descriptor has been proposed [8] which is based on tensor descriptors derived for each of the 8×8 blocks in which the image (and sketch, respectively) is subdivided. For comparing the distances, only blocks are considered in which the user has drawn at least some edge pixels. This descriptor is used by the PhotoSketch system [9], which takes sketches drawn on a graphic tablet with built-in LCD display as

queries to search in a collection of 1.5 million images downloaded from flickr. It allows the composition of new images with the found result images and the help of semi-automated segmentation. Therefore, it does not primarily aim at retrieving known items but also addresses novel item search (and even allows generating novel items; a use-case which is also targeted by the more keyword-oriented Sketch2Photo system [5]). The retrieval quality was compared to the MPEG-7 EHD and turned out to be less sensitive to translation, scale, and rotation. A more thorough evaluation of the tensor descriptor against the Histogram of Oriented Gradients (HOG) [6], ARP, and EHD is presented in [10].

3 Challenges for Known Image Search using Sketches

In what follows, we summarize the most important challenges for a user-friendly system supporting query by sketching.

3.1 Query Input

The first and most obvious step in the process of query by sketching is the generation of a sketch by the user. This imposes the following challenges:

Input Device: Currently, the most frequently used input devices for computers that can also be exploited for the creation of sketches are mice or trackpads. However, they are still not satisfactory for the kind of sketches needed for querying image collections. Thus, more user-friendly approaches need to take into account novel input devices that better support even unexperienced users in providing such sketches.

Color vs. Black-&-White: An important question is whether the image(s) a user searches for contain only black-and-white drawings, full grayscale images, or also use color information. Both grayscale and color information are subjective to the drawing and viewing device, and for the very fine details even subject to calibration of the devices. Selecting the appropriate colors is therefore a non-trivial task and it may also depend on the use case whether or not this information is needed to successfully fulfill the retrieval task. A survey by McDonald and Tait [20] revealed, that users frequently focus very much on color rather than the overall layout of the scene depicted in the image although this does reduce the result quality.

Single Edges vs. Complete Contours: Another important distinction is whether the user has to draw complete contours o objects or only simple strokes of edges of objects. In many cases it might be simpler to draw only some of the most prominent edges rather than to sketch the correct contour outline. For instance, if a real world 3D object is drawn, the latter requires that the user mentally projects the object correctly onto 2D. But to automatically fill parts of the images with color, closed outlines are necessary. When color is used with individual strokes, usually small gaps in between two lines remain. If we consider drawing in arts, it is common to use individual strokes that do not necessarily form closed shapes. Several strokes next to each other are used to give an impression about the intensity of color in that area, even though the individual strokes might have a different color.

Degree of Detail: The time the user is willing to invest in creating a query image is limited. Drawing a very detailed sketch consumes a lot of time, especially compared to browsing result thumbnails. In known item search, the query image will be sketched solely

for the purpose of retrieving images and will be discarded after successful termination of the search process. Therefore, the user might rather trade the time for drawing details in query images for more time browsing the result list. Users may also prefer to iteratively add details and skim the result list to finally stop adding more details as soon as the result list appears to be 'good enough', that is, likely to contain the desired result image among the number of items the user is willing to browse.

3.2 Query Execution

To search for a known item, the user has to recall and sketch this item as good as possible. This not only exacerbates the task for the user, but also makes the execution more challenging since there might be a number of aspects in the query far from being ideal for retrieval.

3.2.1 Empty vs. Unknown Areas

As already described in [13], the sketch of a user is likely to contain large areas left blank. Basically, this can have two completely different meanings: (i.) there is nothing at this spot in the image the user is looking for, just an empty area, or (ii.) the user cannot remember what was at this spot in the searched image, i.e., it is unknown. These two cases are fundamentally different when comparing a sketch to an image, as shown in Table 1. Note that Table 1(a) is symmetric w.r.t. exchanging sketch and image while Table 1(b) is not.

Table 1: Meaning of Area Types in Search

(a) empty areas

Sketch	Image	Meaning
edge	edge	good
edge	space	bad
space	edge	bad
space	space	good

(b) unknown areas

Sketch	Image	Meaning
edge	edge	good
edge	space	bad
space	edge	neutral
space	space	neutral

[13] proposes to set three simple weights α, β, γ for the entire matching process to control the effect of either case. It is possible to have the first case exclusively, so the user does not want any edges except for the areas where they are present in the sketch. However, in case of unknown areas it is rather unlikely that empty spaces next to an edge imply that there can be anything without negative effect. In the extreme case, this would prefer images with high clutter or highly noisy pictures over clean pictures with only some very prominent objects and sharp borders. It is therefore very common to require a *mix of both cases* in the query image as the simple solution does not take this into account.

If there are areas where the user cannot depict the content in the sketch –in particular when she cannot remember– it is likely that the same sketch contains different areas (commonly next to edges sketched by the user) which should be explicitly considered empty, i.e., non-edge areas. It is important to note that there is no general solution to automatically identify which of the two possibilities should be used for a particular area. It is the task of the user to provide the system with this information.

3.2.2 Translation Invariance

It might happen that the user cannot recall where precisely an object / edge was located in the image she is looking for. Thus, this might lead to (minor) differences in the position of single edges or complete objects. Therefore, the system has to provide some robustness against parts which have been shifted. However, complete invariance to translation may also be unwanted since the user may wish to search for images where the sketched object is placed exactly in or at least nearby the chosen area and not somewhere else. Again, a user needs to be able to specify during search whether and how misplacement of the sketch should be tolerated by the system.

3.2.3 Scale and Rotation Invariance

Users, in particular inexperienced users, sometimes have a hard time to estimate proportions when they start drawing. Therefore, even when having a clear picture in mind, they may miss the scale to a certain extent. Hence, a system supporting query by sketching should be able to handle small differences in scale. Very big differences in scale however would usually not occur in known image search but rather when searching for known objects inside (arbitrary) images.

For rotations, the situation is similar: small, accidental rotation may be due to the problem of drawing as well as not remembering precisely the orientation of parts of the image. Additionally, images in the collection may not be stored correctly in landscape or portrait orientation, but rotated by 90, 180, or 270 degrees. This problem occurs less for images taken with digital cameras since many of them nowadays have build-in orientation sensors and store their information in the Exif orientation tags. But for images without such information, like old camera pictures, flatbed scans, or images that lost the information during some step in the image processing chain, this may still be an issue. Thus, rotation invariance will be an essential requirement for searching using user-drawn sketches.

Notice that all these common invariances to affine transformations –scale, rotation, and translation– are essential when images are searched which are not known entirely, for instance when instead of a known image the user searches for novel images which contain certain known parts (e.g., certain objects, people, landmarks, trademarks, etc.) which may appear anywhere in the image. Shearing, another affine transformation, may also be of interest and of course, also the distinction between empty and unknown areas as introduced in Section 3.2.1. Such use-cases therefore significantly differ in terms of their emphasis on these invariances from known image search. Techniques that work well for the tasks of identifying images that contain such items may meet only partially the demands of known image search whereas techniques that satisfy known image search using sketches may certainly not meet all the demands when objects inside images are searched. As this report is targeted towards known image search using sketches, it will not be able to provide also evaluations of these different tasks.

3.2.4 Invariance to Changes in Perspective and Pose

For images of real world objects (i.e., in 3D), the appearance in any 2D projection may change significantly depending on the viewing perspective that was taken when taking or

drawing the picture of the scene. When searching for known images, the user may well remember the individual objects depicted in the scene and the image composition, but will probably not be able to remember the perspective in which the objects occurred — or the user is simply not able to sketch them properly (e.g., by choosing a frontal pose even if the object was turned to a certain degree to the side).

In general, it can be expected a very hard or even unsolvable problem to provide a system that is completely invariant to such changes when all the input it has is a 2D sketch and a 2D image. There are approaches for estimating 3D poses and even complete 3D models from certain 2D images, e.g. [26], but these usually rely on prior information about the images and/or objects and thus cannot be directly applied to a generic image retrieval setting.

If the differences in pose and perspective between sketch and searched image are rather small, the ability that the system will have anyways to cope with artifacts caused by the input device as well as the drawing skills of the user and invariance to small translations, rotations and changes in scale may also partially cover the needs of changes in perspective and pose.

3.2.5 Invariance to Illumination, Color Changes and Focal Length

Due to the different visual appearances between sketches and images, color information is expected to be rather unreliable for comparing the two with each other. Additionally, as described in Section 3.1, the user may prefer to search only on the basis of edge information rather than spending the time to add detailed color information to the sketch.

As long as no light conditions like strong drop shadows or dimly lit or overexposed areas degrade the quality of the edge detection, approaches not using color and relying only on edge information are inherently robust to issues caused by illumination. However, edge detection may be affected by blurred parts of the image, in particular by the choice of the depth of field, one of the most important stylistic devices in photography, caused by the aperture stop selected by the photographer that adjusts the focal length. The user may not be able to guess whether or not the border of an object will still be recognized by the system as an edge. Therefore, she might require a certain robustness against the artifacts caused by edge detection.

3.3 Presentation of Results

In contrast to query by example, the user in a sketch-based approach is able to easily modify the input to the search. When presenting the result, it is therefore desirable not only to show the result images, their ranks and scores, but also give more explanation on what is similar (and what is not). For instance, if search is primarily based on edge information, it should not only be possible to see the real result images, but also display their edge information to the user. Thus, the user will be able to better refine the edges in the query image to the way the system evaluates edge information.

To make best use of the possibility to refine searches, the system has to provide a very interactive user experience: The user must be able to alter the query sketch and search parameters at any time and execute the new search automatically or with little effort. The system must provide the first search results with fast response, in particular when

the search has been modified only slightly as the user may not want to wait as long for small refinements as she would for completely new searches.

Web searches traditionally follow a simple request-response pattern based on the used protocol HTTP and the result presentation in HTML. The user creates the search, clicks on the search button to issue the search request, and the server receives the request, computes the response and the entire result is delivered to the user's browser as an HTML page. Splitting the search results across several pages with a comparably small number of items per page makes rendering the result pages faster — and in case of images in the query result, this limits the number of images that the browser would have to load before being able to show them to the user.

In local desktop applications as well as web applications using AJAX or other techniques for asynchronous result delivery together with the increased memory available, paginating results is no longer necessary. Instead, it might be more convenient for the user to scroll through longer lists. Zoomable User Interfaces (ZUI, [25]) may be used to prevent information overload while still providing quick access to the information needed by the user.

4 The QbS Approach to Query by Sketching

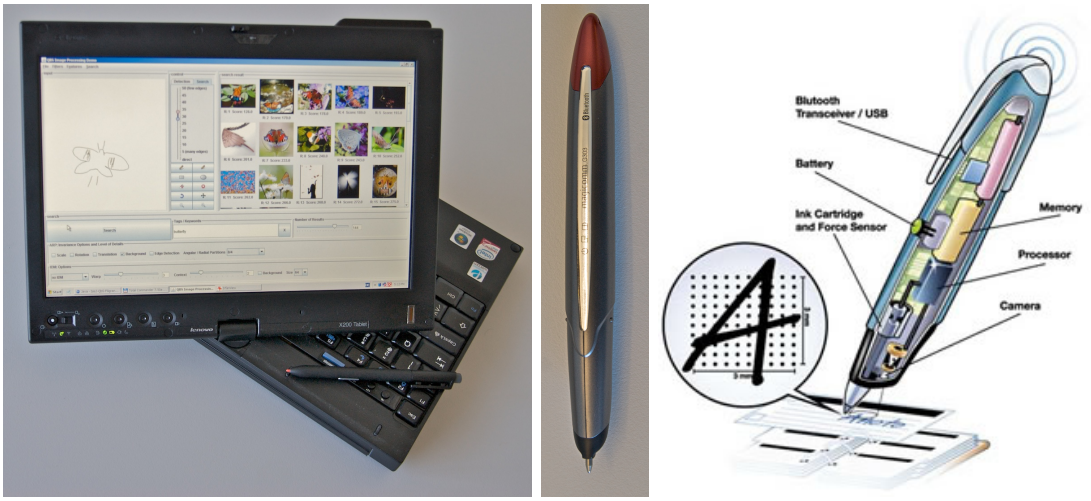
In this section, we introduce the QbS approach to query by sketching and discuss in detail how the challenges listed in Section 3 have been addressed.

4.1 Devices for Creating Sketches

In QbS, users can choose between graphic tablets/tablet PCs and interactive paper to create sketches that will serve as query input.

Graphic tablets or digitizers have been used for many years now, in particular in the domain of CAD. Also digital painting heavily depends on pressure sensitive input devices and therefore relies on graphics tablets. However, novel users usually need quite some time to get used to the hand-eye coordination with traditional graphic tablets that only support the digitization of input without display – so the user is not directly able to see the result of her drawing action. Therefore, even though the QbS system supports graphic tablets, we rather focuses on *Tablet PCs* (see Fig. 1(a)). These devices provide the same functionality for capturing input, but are built into the display of a portable computer. The current interest in touchscreen technology for mobile and surface computing has led to an increasing number of devices that can be used for sketching with or without a stylus.

In addition to tablet PCs, QbS focuses on novel *interactive paper and digital pen* interfaces [12] that have been introduced in 2003. These devices allow to capture a drawing on paper and transfer it to a computer without the need of scanning the original paper. A special pattern printed on the paper and a small camera inside the digital pen as depicted in Figures 1(b) and 1(c) enable the pen to recognize the position on the paper. Via a USB or Bluetooth interface, the pen can communicate directly with a computer. Dedicated areas on the paper can also trigger certain actions, such that paper becomes an input device that can also issue commands [22]. With these devices, QbS offers a completely new user experience for sketch-based CBIR since they allow the user to draw on regular paper just as they would do if no computer was involved [30].



(a) (b) (c)

Figure 1: Novel input devices for sketching: Tablet PC (a) and digital pen for interactive paper (b,c).

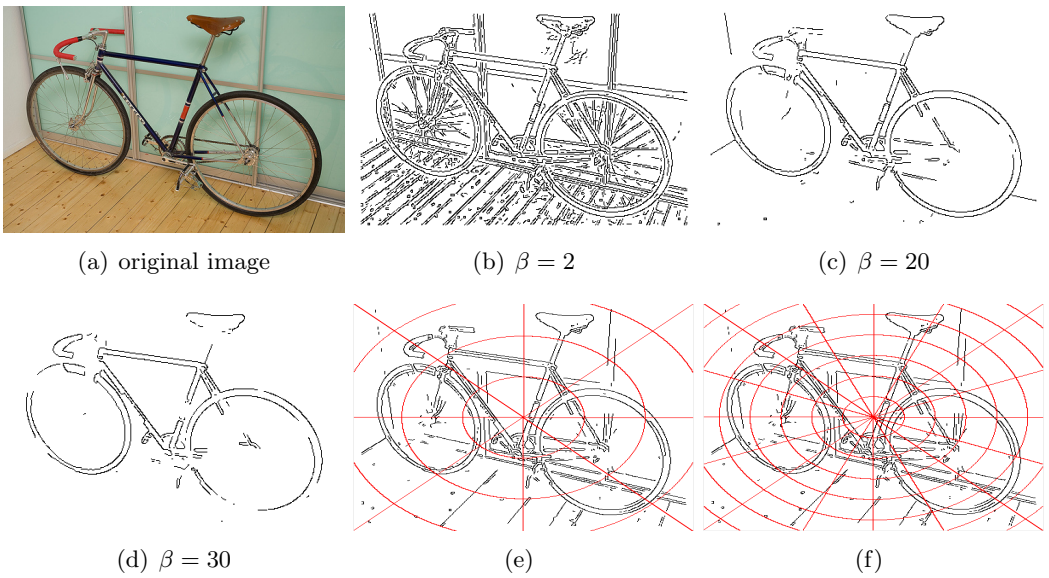
4.2 Retrieval Process

Using one of the devices described above, the user starts to draw some edges, but not necessarily closed contour outlines. The QbS system provides a fast query mechanism to retrieve some of the top-ranked results to give the user a fast feedback about the retrieval result that can be expected from the final or full result. Then, the user is able to add further details to refine the search and remove misleading parts of the sketch without having to redo the complete sketch.

In contrast to directly comparing images with each other –as it is the case in CBIR– sketches and images always have to receive some preprocessing in order to make them comparable. This also limits to a certain extent the features that can be used. In order to properly deal with the invariances described in Sections 3.2.2–3.2.5, either the features themselves must provide this property or a particular distance function is needed. The distinction between empty and unknown areas introduced in Section 3.2.1 can only be achieved by means of appropriate distance functions.

The study in [20] has shown that query by sketch is able to assist the users in a best possible way if it is used in cases where the user has seen the item before search. Therefore, she will not only be able to provide the system with the sketch as input, but also some additional information on the invariances that have to be considered by the system and on how to distinguish between empty and unknown areas.

QbS focuses on edge information and does not rely on semi-automatic segmentation or annotations of the image database, in contrast to QBIC's shape features or MPEG-7 CSS — since the latter usually require significant work at the time of inserting images to the collections which users are frequently not willing to invest. Furthermore, QbS is based on the use of edge maps as defined in [3] for the use with Angular Radial Partitioning (ARP). Sketches as well as images in the collection available for search are examined at a resolution of 400×400 pixels, which is a much higher resolution than QVE or IDM would



(a) original image (b) $\beta = 2$ (c) $\beta = 20$

(d) $\beta = 30$ (e) (f)

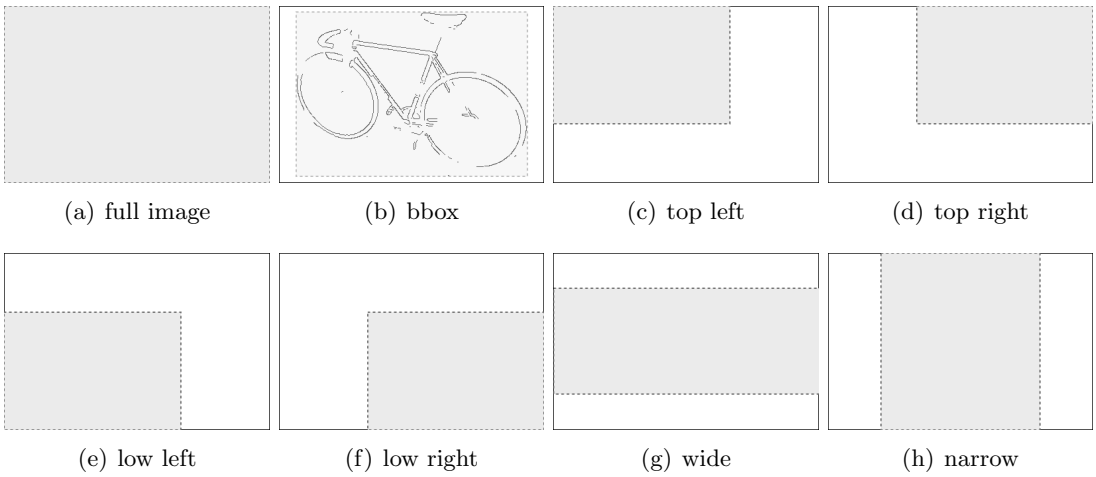
Figure 2: An example image (a), edge maps for various values of β (b-d), partitioning with $\beta = 10$ exploiting 8 angular and 4 radial partitions (e) and 16 angular and 8 radial partitions (f).

commonly use. The generation of edge maps is controlled by the threshold value β (c.f. Section 2) as illustrated in Figure 2, where low values preserve many edges while high values retain only very few and prominent edges. To better compensate for effects due to level of detail (c.f. Section 3.1) and edge detection (Section 3.2.5), we do not define a single value of β for the entire collection, but rather extract the edge maps from the images at several values between 2 and 50.

We incorporate two different sets of features and corresponding distance measures. First, Angular Radial Partitioning (ARP) is used as a compact, fast way to retrieve images when rough sketches and spatial layout is sufficient to separate the desired known item from all other images in the collection and robustness against many invariances are needed due to the deviations of the sketch w.r.t. to the known item. Second, an adapted version of the Image Distortion Model (IDM) on the same edge maps is used as a more complex, computationally more expensive solution whenever the user needs a more thorough comparison between the sketch and the other images. For IDM, the user has to provide a sketch that is detailed and located close enough to expect meaningful results. In addition, in both cases the user can restrict the comparison to images selected on their metadata, e.g., only consider images that have been tagged with certain keywords.

4.2.1 Similarity using ARP

We support ARP at various resolutions. By default, ARP is used with 8 angular and 4 radial partitions (as depicted in Figure 2(e)). In this case, the number of edge pixels inside the sketch and the edge map are counted, which will generate a 32-dimensional feature vector. In order to distinguish empty from unknown areas as defined in Section 3.2.1, the vectors are compared with a weighted Manhattan distance. In the easiest case of treating



(a) full image (b) bbox (c) top left (d) top right

(e) low left (f) low right (g) wide (h) narrow

Figure 3: Image Regions

all non-edge areas as empty areas, all weights will be set to 1 which is equal to using an unweighted Manhattan distance. In another simple case where partitions without any single edge pixel are treated as unknown areas, such partitions of the sketch will cause a weight of 0 and otherwise 1, therefore ignoring the any deviation between the sketch and compared images for this partition while still summing up the absolute differences in counted edge pixels in all non-empty partitions. The first case is the only case considered in [3], the latter case is equivalent to the approach proposed in [8] for a completely different descriptor.

The user can choose between the two cases, and therefore has the freedom to either sketch the full image or to focus on drawing only the parts of the image she remembers well, treating all other areas as unknown. The approach can be easily extended to let the user also define explicit relevance of areas as this information can be incorporated into weights on partitions. Such regions of interest [31] can be easily selected using input devices mentioned in Section 3.1.

In particular, when only parts of the image are considered relevant, it becomes important to also ask the user whether she assumes the sketch to be placed in the appropriate position or request from the system to be less sensitive to translation. In cases where the entire image is used to define empty areas around the searched sketch, translation invariance might be important, for instance when searching for trade marks. As long as not too many edge pixels move to a different angular radial partition, ARP will not be very sensitive. For translations that are further off the position of the sketch, a heuristic can be applied: the original image is cut into several regions that still cover enough pixels to give meaningful search results. The ARP features are extracted for each of these regions (Figure 3(c)-(f)), as well as for the image as a whole (Figure 3(a)) and a bounding box on all non-empty pixels (Figure 3(b)). During search, the sketch will be compared to all these regions, thus compensating to a certain extent for translation. As most of these regions are smaller than the full image, comparing the sketch or parts of the sketch can also provide some invariance to scale. Additional regions for different aspect ratios (Figure 3(g)-(h)) assist when the scaling does not maintain proportions.

Small degrees of rotation will also not shift too many edges from one partition to another, such that ARP will not be very sensitive to such deviations either. However, if the user expects even more rotation, invariance to rotation is provided by applying the 1D FFT on the features as proposed in [4].

Since the ARP feature vector is rather compact, it is feasible to extract and store several variations of parameters in order to enable invariances. Therefore, multiple sub-images for translation invariance and aspect ratios are processed and the 1D FFT for rotation invariance is applied and stored separately for each chosen β value. For simple, regular searches the query feature vector from the user's sketch will be compared to exactly one feature vector for each image in the collection. Invariant searches will compare the distances with all corresponding representations, but only select the best-matching version of the image features for an individual image to compute its distance.

4.2.2 Similarity using IDM

For IDM, the edge map image is scaled down to smaller sizes. As default size, we use 32 pixels for the longer side. Since the user is expected to draw the edges in the sketch, edge detection is never applied to the sketch. This means that the scaling has to be performed after the edge detection. Moreover, it is preferable to use an interpolation for scaling down and to apply thresholding afterwards to return to a binary image (edge / non-edge) rather than not using interpolation in scaling, which may drop a lot of the edges.

The user can parameterize the search with IDM to fit her needs very precisely. With the warp range, the user can specify how many edges are allowed to be misplaced by translation, scaling, or rotation. This misplacement is measured in terms of the number of pixels in the reduced resolution of the image, e.g., at most 32 pixels on the longer side. Big warp ranges result in a higher invariance. A warp range of 3, for instance, would allow an edge to be misplaced by 3 of the at most 32 pixels in every direction which is roughly +/-10%. The size of the local context defines whether individual pixels (local context of 0) or patches (local context > 0) are matched. Big local contexts result in small invariances as bigger patches must fit to achieve low distances. A local context of 2, for instance, would describe a patch of size 5×5 pixels.

To deal with invariances w.r.t. the value β used in extraction, the same strategy as for ARP is used. And also similar to ARP, unknown areas can be handled by ignoring non-edge pixels in the scaled down sketch. They only affect the distance score as part of the local context of an edge pixel — and more complex relevance judgements of areas in the sketch can be easily integrated as weights.

To reduce execution time, we use the early termination strategy described in [29] for IDM, but also in the computation of the Manhattan distance needed for the ARP features. The execution costs are of course still much higher than for ARP, but through the choice given to the user, she can decide whether to use the ARP features and get results almost instantly, or use IDM when she thinks her sketch is very detailed and will lead to very precise results. Another option is to initially start with ARP to get a set of images that can then be re-ranked according to IDM. This scenario avoids an exhaustive similarity search via IDM on the whole collection, but rather on a smaller subset retrieved by ARP.

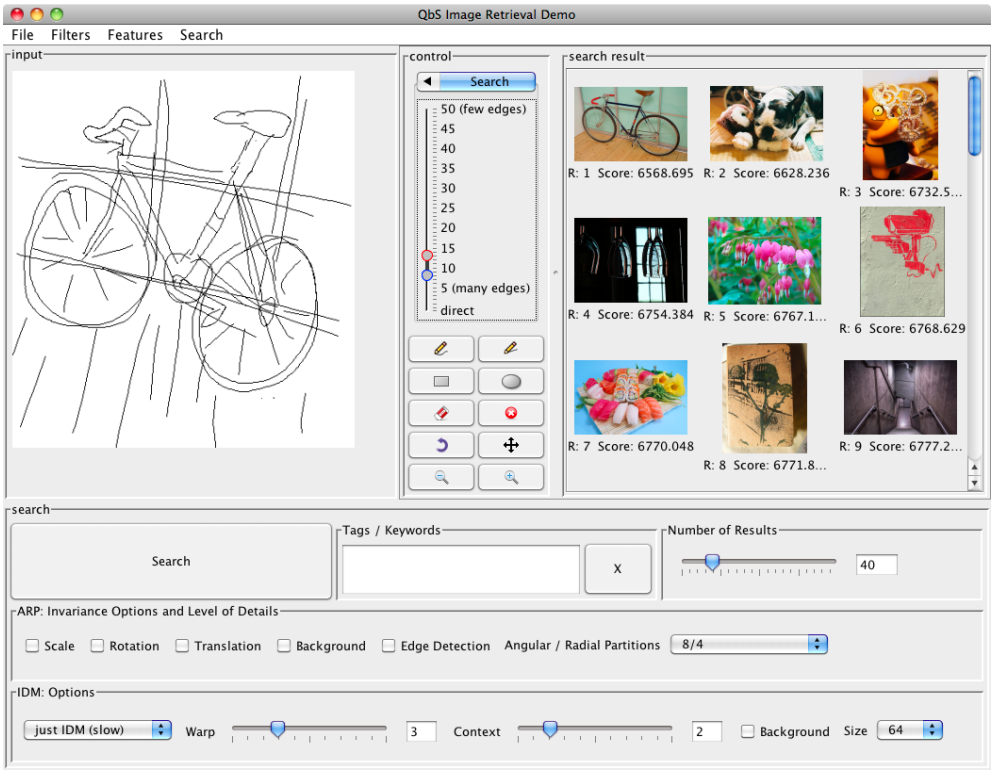


Figure 4: Search with User-Generated Sketch

4.2.3 Integrating Filters

If the images are attributed with metadata (e.g., tags or keywords) and the user remembers them, search can be reduced to images that contain these matadata. The easiest way to support this is to perform a sequential scan over the features and skip all features of images that do not satisfy the constraint given by the user. Performance can be further improved by creating an index to look up all images that satisfy these constraints.

However, it is important to notice that in most cases the images in the collection are not attributed with (good) keywords. In particular, private image collections like the pictures taken with digital cameras which are not shared online are frequently lacking such metadata. Even for resources that are shared online, there is often no shared or controlled vocabulary such that the user searching for one particular image may simply not know the 'right' keywords to find an item in such a collection. [2] showed also that not all tags can be used for search. Therefore, it is important that the QbS approach does not require such information to achieve satisfactory results but is able to exploit it, if available.

5 The QbS Prototype

The user interface of the QbS system is shown in Figure 4. The user can draw a sketch on the left part of the screen, and result images are presented to the right. The search button and options are located in the lower part of the screen.

For ARP, the user can simply tick the checkboxes of invariances that should be enabled. For IDM, the user can specify the desired warp range and local context. The slider in the middle allows to define a range for the β value for edge detection. Enabling invariance to edge detection is equivalent to selecting a range over all values.

Search is performed by applying a k-nearest neighbor sequential scan over the file(s) containing the features. Since the ARP features are rather compact, caching the features in memory is not a problem on current hardware, not even for thousands of images. To handle the possible combinations of features, a simple Least Recently Used (LRU) strategy is used to keep subsequent searches fast, allowing interactive modifications of the sketch before re-submitting a query. If query execution time should still be crucial, a high-dimensional index structure such as, for instance, the VA-File [32] can be used after modifications of distance evaluation to make sure that invariances in the retrieval process (cf. 4.2) are supported by the index.

IDM features are considerably larger than ARP and in particular query evaluation with IDM is always significantly slower than ARP – even when the early termination strategy [29] and multi-threading is applied. In order to reduce the required time, ARP can be used as a filter to select candidates and IDM can be used only to re-rank these candidates in order to get the final results. But the number of candidates subject to re-ranking is crucial in the case of known item search: The number of candidates returned by ARP must be big enough to contain the known item – otherwise no improved ranking will ever be able to help the user to find the desired item.

As mentioned in Section 4.2.3, the search can be limited only to images that have been tagged with particular keywords. This also radically reduces the time needed for similarity computations and therefore improves the search time for ARP as well for IDM. In QbS we use Lucene[4] to build a full-text index.

Because of fast query execution, in particular in the case of ARP, loading images from disk can become the bottleneck in user experience. To be able to present results quickly, previously retrieved thumbnail images are held in an LRU cache (similar to the features), such that only slightly modified searches will display results almost instantly while newly retrieved items are loaded in a background thread from disk.

If the user left-clicks on a thumbnail in the result list, the original image is presented in the size of the drawing area. After right-clicking on a result, the edge map of the image is presented as depicted in Figure 5. This allows a user to easily compare the sketch with the retrieved results and to modify the query parameters, e.g., refine the sketch or enable/disable options for invariances. The smaller images shown in an overlay in the lower right of the sketch/edge map give a visual representation of the features used for IDM. For ARP, an overlay of the partitions can also be enabled.

6 Evaluation

We use the MIRFLICKR-25000 [5] dataset [14, 15] for our evaluations, a collection with 25,000 images that were downloaded from the popular image sharing website flickr.com. The collection consists of photographs that have been selected based on their license (only

[4] `http://lucene.apache.org/`

[5] `http://press.liacs.nl/mirflickr/`

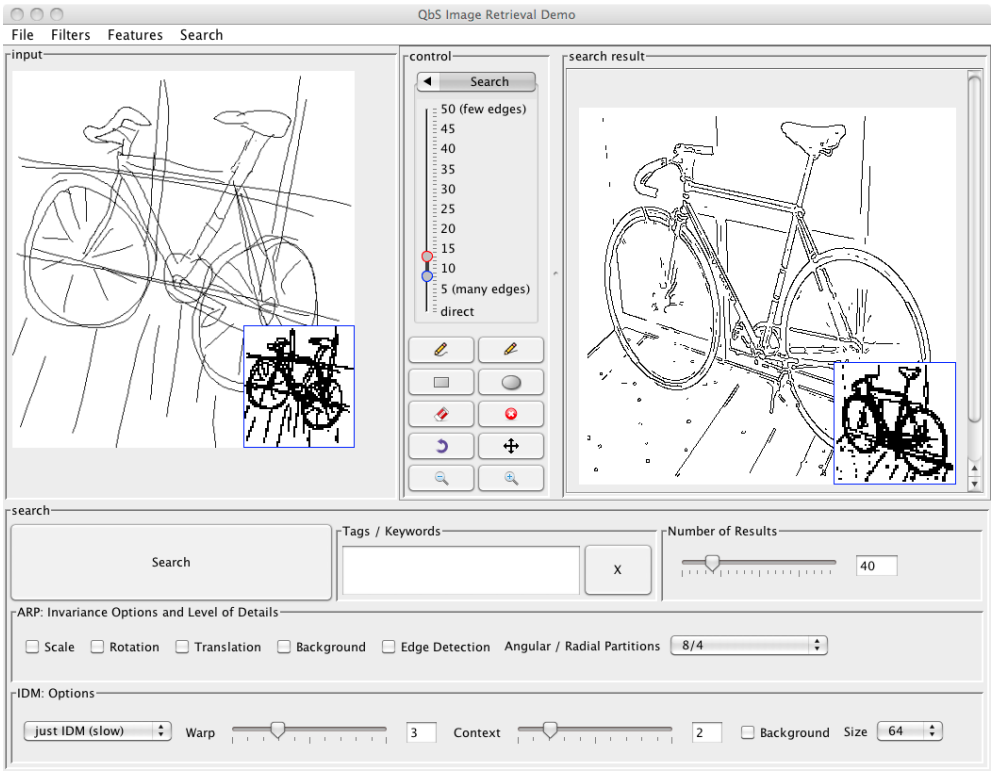


Figure 5: Sketch and Edge Map of the Known Item with IDM Overlay

Creative Commons-licensed pictures) and on the flickr measure of "interestingness" which takes into account how many people watched the image, commented on it, tagged it, picked it as favorite on flickr at the time when the collection was created. The images in the benchmark collection are accompanied with metadata on the photographer, the title and tags of the image, Exif metadata, and license information.

6.1 Performance Measures

Throughout our evaluation, we use the rank as the main measure to assess retrieval performance of QbS. There are other commonly used statistical measures in evaluating information retrieval systems like precision, recall and the Average Normalized Modified Retrieval Rank (ANMRR) [19].

We will discuss in the following subsections why we decided on the plain rank as the measure of choice for measuring the effectiveness of QbS and why we did not rely on widely used measures like precision, recall, and the ANMRR.

6.1.1 Precision & Recall

Precision and recall [27, p. 164 et seqq.] are very well established measures for the retrieval quality. Precision is the measure of the ability of the system to present only relevant items. More formally, it specifies the number of relevant items retrieved, divided by the overall number of items retrieved by the system. On the other hand, recall is the measure of

the ability of a system to present all relevant items. More formally, it is the number of relevant items retrieved, divided by the number of relevant items in the entire collection. Table 2 classifies the resulting query items into four sets according to their relevance and retrieval state. Consequently, equations (1) and (2) present precision and recall in set representation.

	Relevant	Irrelevant
Retrieved	A	B
Not retrieved	C	D

Table 2: Classification of Query Results according to Relevance & Retrieval

$$\text{precision} = \frac{\text{A}}{\text{A} \cup \text{B}} \tag{1}$$

$$\text{recall} = \frac{\text{A}}{\text{A} \cup \text{C}} \tag{2}$$

We avoid using these two measures in our evaluation as we are evaluating the search for known items, which implies that for every query the cardinality of all relevant items in the entire collection is always $|A \cup C| = 1$. Therefore, the only possible values for recall will either be 1 (if the known item is retrieved) or 0 (if the known item is not found), which does not provide sufficient information to evaluate the performance of an information retrieval system. Precision and recall are set-based measures that are used to evaluate the quality of an unordered set of retrieved documents. To interpret both notions in combination and to get a feeling for the quality in ranking, precision is commonly plotted against recall resulting in a precision-recall graph. Trying to plot this graph for our single known item search will result in points being drawn at either recall of 0 or 1.

Another related measure is the Mean Average Precision (MAP) which provides an overall measure of the quality across all recall levels. It is calculated by averaging the precisions computed at each of the relevant items in the ranked list. In our case of known item search, there is only one relevant item. Therefore, calculating MAP is not helpful either.

6.1.2 Average Normalized Modified Retrieval Rank

Another popular measure developed by the MPEG-7 research group to evaluate information retrieval systems is the Average Normalized Modified Retrieval Rank (ANMRR) [19]. In this measure, let NG(q) represent the number of ground truth images for a given query q. Let K represent a cutoff number that specifies the maximum rank that would count as feasible. We will give K the value of 25,000 since we are attempting to search for the known image within the entire MIRFLICKR-25000 collection and we are interested in the rank obtained even if it is relatively poor. The Rank(k) is the retrieval rank of the ground truth image k that is retrieved within the top K items in the entire collection. MPEG-7 penalizes ranks that exceed K as shown in Equation 3. However, since we have chosen K

to be the size of the entire collection, no rank will actually exceed K. Therefore, only the first case of the equation will become effective.

$$\text{Rank}^*(k) = \begin{cases} \text{Rank}(k) & \text{if Rank}(k) \leq K(q) \\ 1.25K & \text{if Rank}(k) > K(q) \end{cases} \tag{3}$$

The average rank AVR(q) for query q is computed as follows:

$$\text{AVR}(q) = \sum_{k=1}^{\text{NG}(q)} \frac{\text{Rank}^*(k)}{\text{NG}(q)} \tag{4}$$

In the case of single known item search, NG(q) will always be equal to 1. Therefore, AVR(q) will simply be equivalent to Rank(k). The modified retrieval rank (MRR) is formulated to minimize the influence of variations in NG(q). It is calculated as follows:

$$\text{MRR}(q) = \text{AVR}(q) - 0.5 * [1 + \text{NG}(q)] \tag{5}$$

Nevertheless, in known item search for a single item, it will simply decompose to Rank(k)-1. Hence, MRR(q) will give a value of 0 in the case that we retrieve the known item at the top of the list. However, the upper bound of MRR(q) still depends on NG(q). To normalize this value, the Normalized Modified Retrieval Rank (NMRR) is defined as follows:[6]

$$\text{NMRR}(q) = \frac{\text{MRR}(q)}{1.25K - 0.5 * [1 + \text{NG}(q)]} \tag{6}$$

Therefore, in our case of single known item search on the MIRFLICKR-25000 collection, NMRR(q) will decompose to the following equation:

$$\text{NMRR}(q) = \frac{\text{Rank}^*(k) - 1}{1.25 * 25000 - 0.5 * [1 + 1]} = \frac{\text{Rank}^*(k) - 1}{31250 - 1} \tag{7}$$

The NMRR will always be in the range of 0 to 1 and the smaller the value is, the better is the retrieval quality. In our case, NMRR(q) for single known item search will give 0 if the known item is retrieved at the top spot. On the other hand, it will give 1 to indicate that the item was not found and the rank expresses the penalty defined in Equation 3. For our case with $K = 25000$, a value of $24999/31249 = 0.8000$ would express that the item was found at the worst possible rank.

Finally, to compute the Average Normalized Modified Retrieval Rank (ANMRR), NMRR of individual queries is averaged over all queries with NQ being the number of queries:

$$\text{ANMMR} = \frac{1}{\text{NQ}} \sum_{q=1}^{\text{NQ}} \text{NMRR}(q) \tag{8}$$

[6]The intention of the divisor is to be the worst rank possible, therefore normalizing the results to the range $[0, 1]$. As Equation 3 is not used / needed when K is as big as the dataset itself, the factor 1.25 in the divisor would be obsolete. It is just kept in order to stay in line with the MPEG-7 definition of ANMRR.

Therefore, as explained with NMRR, ANMRR will always be in the range of 0 to 0.8000 for known images from the collection when searching until the item is found.[7] Consequently, in our case of known item search for a single item, reading results containing ranks of items directly will be much more understandable, e.g., average ranks of 1.2, 6.33, 15.43, 52.4, 118.67 are more readable than the corresponding ANMRR of 0.0000064, 0.00017, 0.00046, 0.00164, 0.00377.

6.1.3 The Rank

In our evaluations, we decided to rely on the rank of the retrieved known item as a measure of the system, and since we search for each known item using different sketches we calculated the following:

- Best rank
- Worst rank
- Mean rank
- First quartile
- Third quartile

A compact form of this information is presented in Figure 11.

6.2 Preparation

In preparing for our evaluation, we needed to select a set of images to act as the known items which the users had to search. We chose four different images that vary in characteristics with respect to difficulties in drawing a sketch of them. We presented these images in printed form to a group of seven people with different sketching abilities and collected the resulting sketches. Furthermore, since one objective of the evaluation is to also assess the use of text search accompanying ARP and IDM, we selected a couple of keywords from the flickr tags that users associate with the four images. Details about these preparation steps are explained in the following subsections.

[7] For comparison: If the ANMRR of random results is computed with K being all images of the collection, one would expect an average NMMR(q) of about 0.5, but due to the penalization factor 1.25 it becomes $12500/(31250-1) = 0.400$. In contrast, if K would be set e.g., to 1000, we would expect Rank$^*(k)$ to return 1250 for random results in 24,000 of 25,000 cases in known image search and as a consequence an ANMRR of $(500 * (1000/25000) + ((1250-1) * (24000/25000)))/(1250-1) = 0.976$. For $K = 100$, the ANMRR would be 0.9976.
This shows that the interpretation of ANMRR in the context of known image search always depends on the choice of K with respect to the overall collection size. While 0.4 might be a good result for very small values of K as it significantly outperforms random results, the same value becomes very bad for big values of K. Such a problem in interpretation does not exist when either only small values of K are used (but this does not represent well the task when the user can only stop searching once the item was found), or the rank is used directly. Notice that there are attempts to define reasonable values of K for the use with ANMRR, e.g., that K should be at least twice as big as the number of ground truth images for a single query $NG(q)$ as proposed in [19, 4], but for small sets of ground truth images and in particular the extreme case of known item search with $NG(q) = 1$, more tolerant values of K are needed.

(a) im1660.jpg (b) im10853.jpg (c) im18707.jpg (d) im18797.jpg

Figure 6: Images selected for evaluation

6.2.1 Selection of the Known Items for Evaluation

We have performed known item search on four images that have different characteristics with respect to difficulties in drawing a sketch of them.

im1660.jpg in Figure 6(a) shows basically a single object, which is located in the center of the image. Moreover, the background is rather homogeneous and different from the foreground, which results in the ability of edge detection to find some value of β where all of the edges in the background are removed while still many edges of the foreground are preserved. In this particular case, at a value of β=7, none of the clouds contribute to edges anymore while almost all details of the plane are still available. This can be seen by examining the edge maps generated at various β values in Figure 15. Such characteristics makes it comparably easy to retrieve this image from a sketch as the common intuition of users to draw just the desired object works very well and also placing an object directly in the image center is easier than estimating a displacement from the center. As the background disappears even for comparably low values of β, a bounding box region (as in Figure 3(b)) can also compensate for translation and scaling.

im10853.jpg in Figure 6(b) is considerably more challenging as it contains several distinct objects of interest (e.g., houses, mountains). There are plain or homogenous areas like the sea or the sky as well as areas with strong contrasts at high frequency like the rocks and vegetation on the mountains. The spatial distribution is not very hard to estimate and draw, however there might be issues with placing the coastline too high or too low and scaling, in particular matching the aspect ratio and relative sizes of objects.

im18707.jpg in Figure 6(c) contains again one main object, but this time, it is rather difficult to separate it from the image background. As can be seen in Figure 17, there is no value of β at which the details of the bike would still be preserved while the floor and walls would already disappear. Notice also that segmentation in image processing as well as the classification of bicycles based on visual features in the area of pattern recognition is considered challenging due to the property that most of the object's area does not block the line of sight to the background. The image perspective is also non-trivial as there is no planar view on the bike as e.g., a frontal or lateral view.

Finally, im18797.jpg in Figure 6(d) contains several prominent objects, although the hand and figure of the Eiffel tower are dominant. These objects are neither placed directly at any image border nor directly in the center of the image. Significant parts of the image are blurred, but colors still differ too much to generate areas which would appear to edge

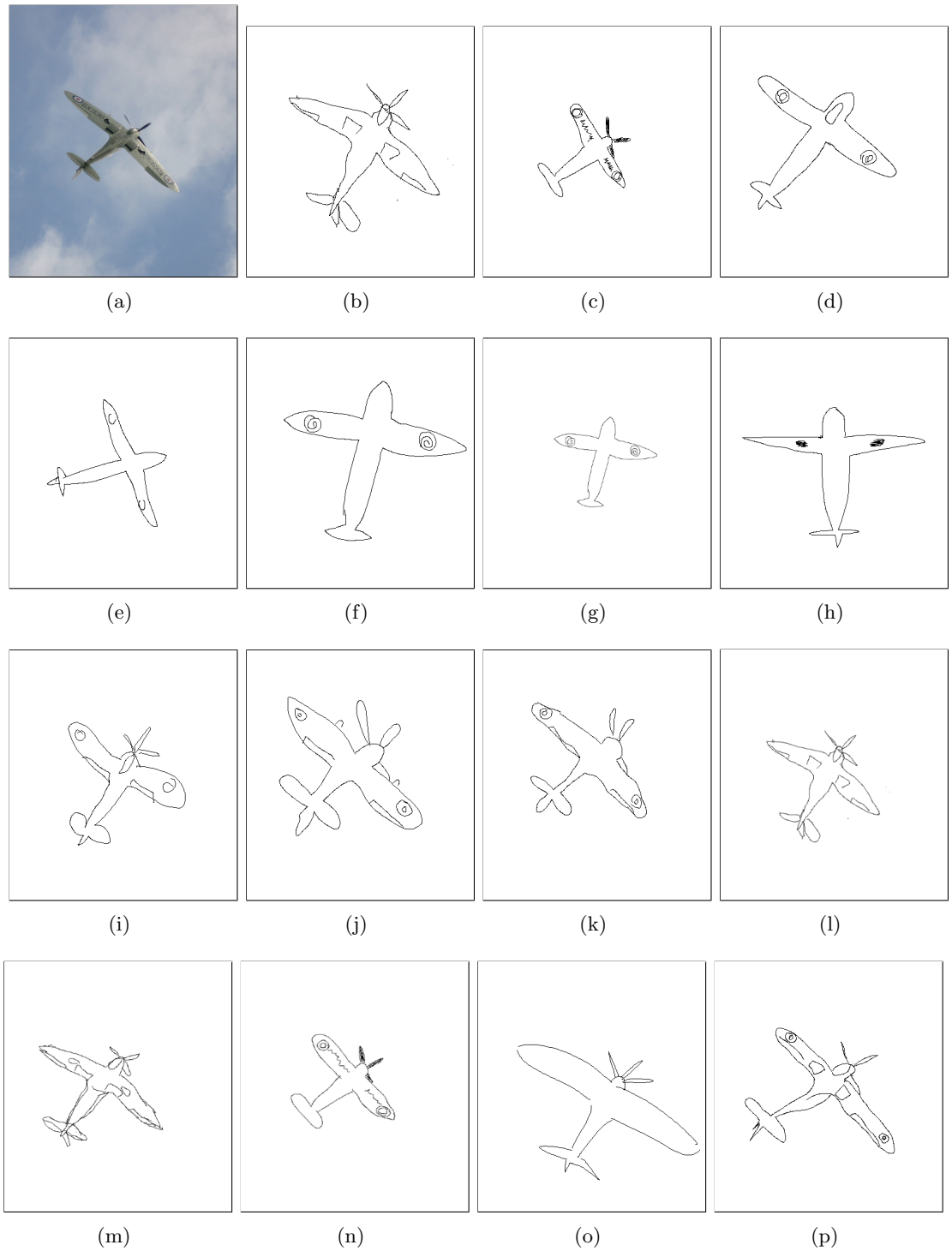
detection as homogenous areas. There is some (almost) empty area in the lowest part of the image which commonly makes it harder to place the non-empty areas in correct proportion to it in the sketch. Finally, none of the major visible lines are either horizontal or vertical; all are rotated a bit against those orientations.

6.2.2 Sketch Acquisition

For the purpose of evaluation, we have collected sketches for these four images from a group of seven people with different sketching abilities. The individuals were given time to familiarize with the QbS system running on a Lenovo X200t Tablet PC system as shown in Figure 1(a). Furthermore, they were then requested to search for the known items which were shown to them in printed form. We collected a total of 15 sketches for im1660.jpg (Figure 6(a)), 14 for im10853.jpg (Figure 6(b)), 15 sketches for im18707.jpg (Figure 6(c)), and ten sketches for im18797.jpg (Figure 6(d)). All sketches obtained are shown in Figures 7, 8, 9, and 10.[8]

Out of the four images, none was the only image either showing that kind of object in this collection or the only one using this particular kind of composition. As we use features that do not take color into account, the results are also not biased due to the selection of images using exceptional colors or color distributions.

[8] All sketches used in the evaluation are available for download on our website: `http://dbis.cs.unibas.ch/downloads/qbs/QbS-User-Sketches.zip/view`



(a) (b) (c) (d)

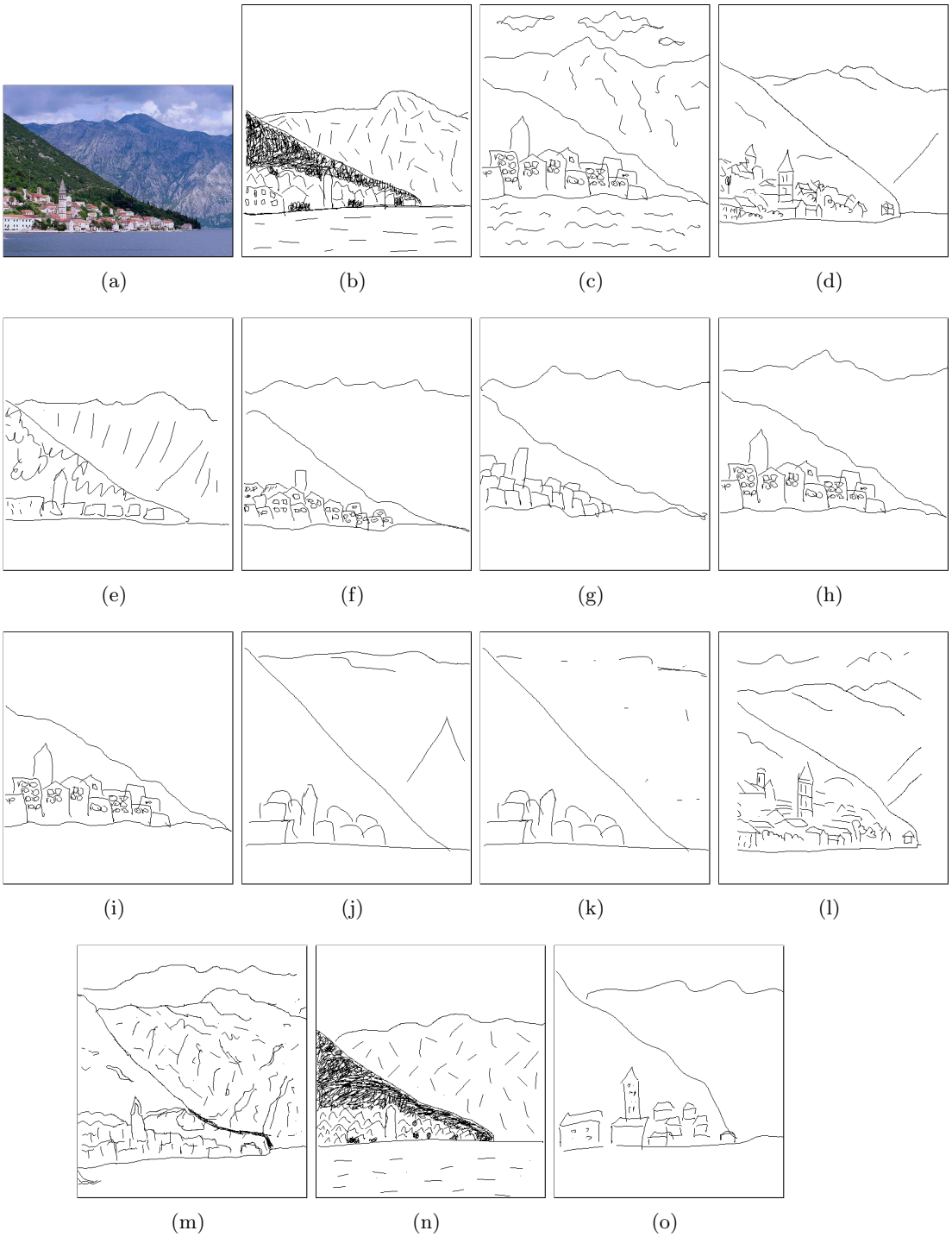
(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

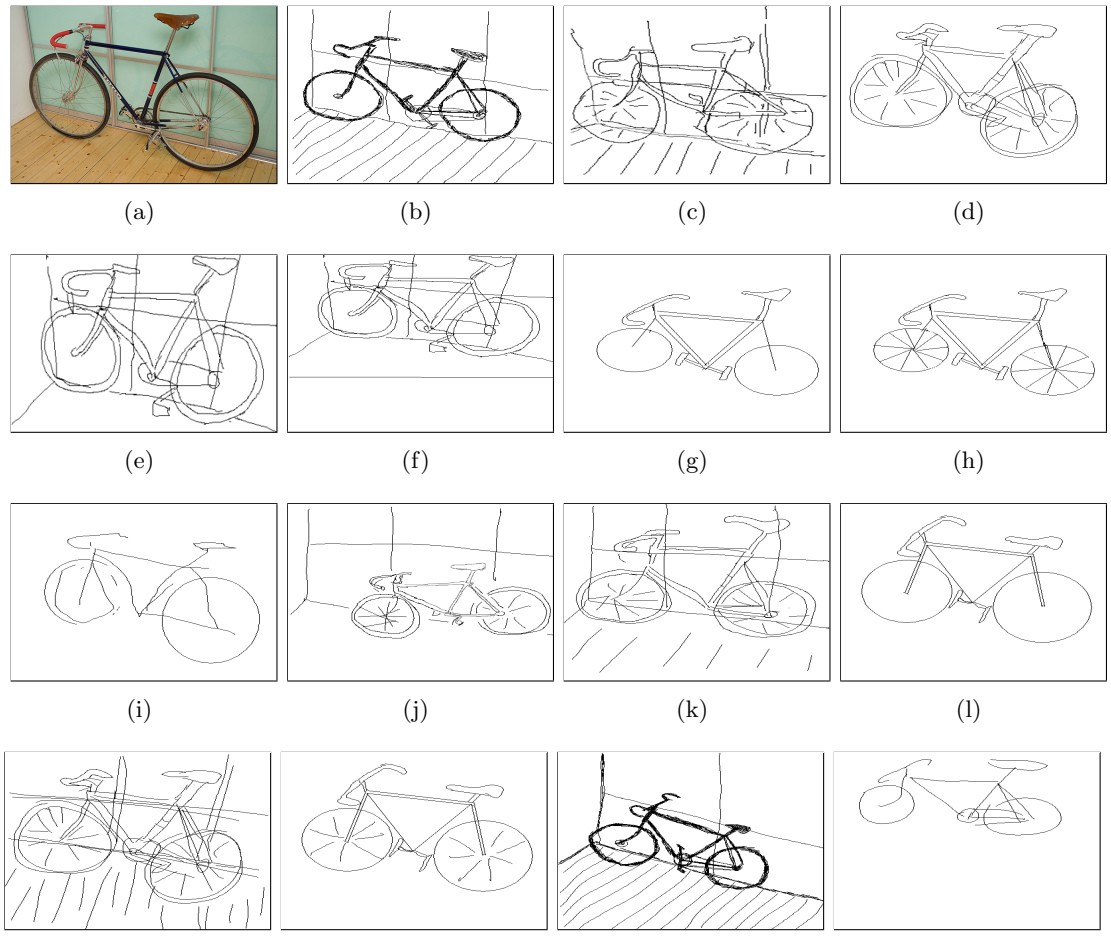
im1660.jpg: 'Supermarine Seafire MKXVII' by Alex Layzell, License: CC-BY-NC-ND

Figure 7: Sketches for image im1660.jpg used for evaluation.



im10853.jpg: 'Perast - Montenegro' by Milachich, License: CC-BY-NC

Figure 8: Sketches for image im10853.jpg used for evaluation.



(a) (b) (c) (d)

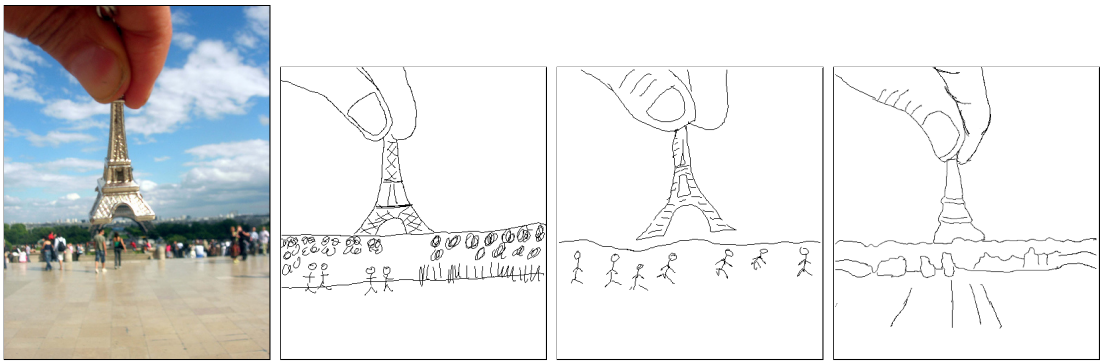
(e) (f) (g) (h)

(i) (j) (k) (l)

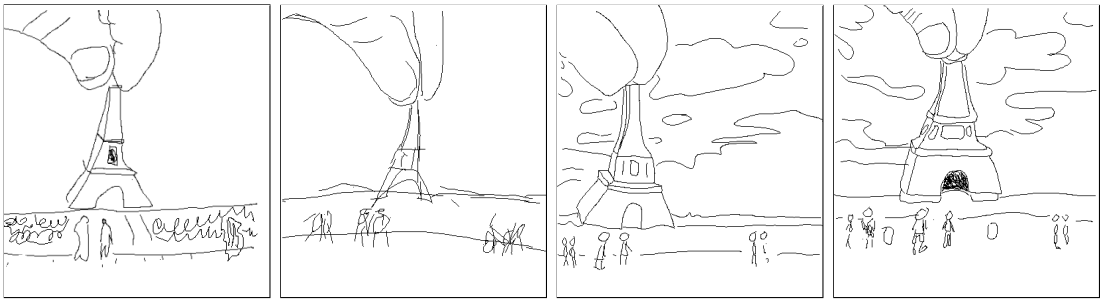
(m) (n) (o) (p)

im18707.jpg: 'Version 1.0 release candidate 1' by Petteri Sulonen, License: CC-BY

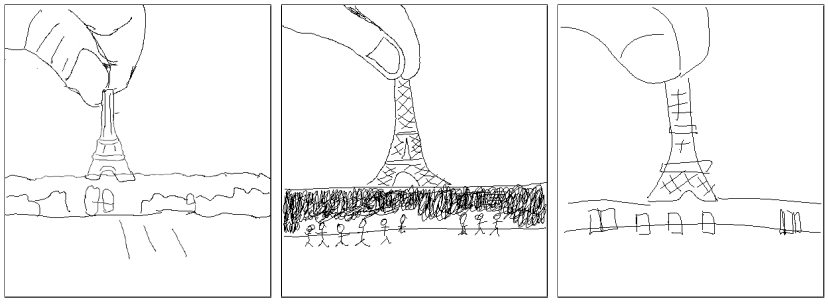
Figure 9: Sketches for image im18707.jpg used for evaluation.



(a) (b) (c) (d)



(e) (f) (g) (h)



(i) (j) (k)

im18797.jpg: 'If Eiffel In Love With You' by Gideon, License: CC-BY

Figure 10: Sketches for image im18797.jpg used for evaluation.

6.2.3 Selection of Tags for Known Items

We measured the individual performance of ARP and IDM. In addition, we assessed the use of text search accompanying each algorithm. For the keyword search, we used the two most appropriate tags that flickr users associated with the four images. The selection of the keywords was performed with feedback that we received from the people contributing sketches when we showed them the list of tags associated with the images. Through this, tags that were highly specific like "singlespeed" for im18707.jpg (Figure 6(c)) and "trocadero" for im18797.jpg (Figure 6(d)) have not been selected, even though they would have been appropriate – as their relation to the image where unknown to most of the users. The complete selection of tags that were chosen in our evaluation are presented in Table 3. It is worth mentioning that when the tags where presented to the people participating in the study, they were presented in the original order of the MIRFLICKR data set. The tags displayed in Table 3 are ordered, having the least specific tags on top. This means that the tags that have been assigned to most images are listed first, the tags that would filter out most images are listed as last. This order has been chosen as it nicely shows that the more specific tags are never selected by the users — they are too specific and therefore unknown to people who did not tag these images themselves.[9]

Table 3: Selected Tags used in Text Search Filtering

Image	FLICKR Tags (Selected Tags are highlighted in **bold**)
im1660.jpg	canon, 2008, usa, spring, uk, old, europe, digital, eos, england, museum, rebel, world, **flying**, war, us, aircraft, show, flight, **plane**, air, 2, britain, engine, military, 300d, display, arm, airshow, royal, gb, wwii, east, ww2, british, united, english, fighter, force, aerobatic, raf, ii, european, aeroplane, duxford, kingdom, aerobatics, imperial, fleet, iwm, supermarine, seafire, trainer, piston, warbird, airworthy, anglia
im10853.jpg	blue, water, beach, **sea**, sun, sand, life, fun, **mountain**, beauty, montenegro, milachich, balkan
im18707.jpg	vintage, **bike**, **bicycle**, fixedgear, singlespeed, fixedwheel
im18797.jpg	**paris**, ring, **eiffeltower**, key, trocadero

[9]This experience is important as in cases where users search for their own images with their own tags or keywords, e.g., for personal image collections, they may use more specific terms and therefore get better search results using text retrieval. However, personal image collections or other collections that are not shared between people are commonly annotated more sparsely than the (selected) images that are shared and that can be annotated by several people by exploiting the "wisdom of the crowd".

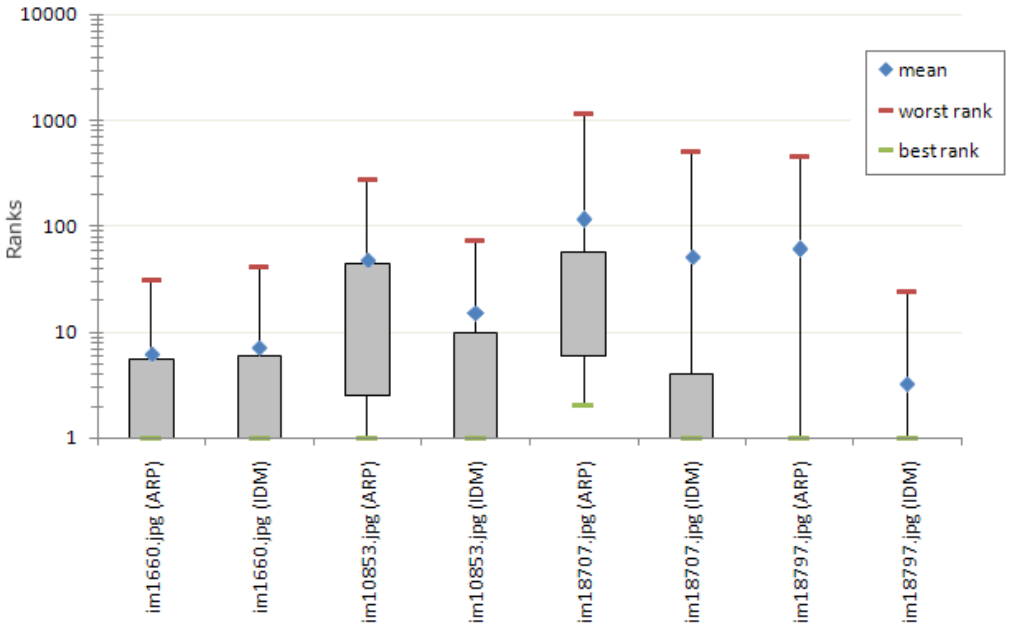


Figure 11: Ranks of Known Items (Text Filter off)

6.3 "Brute Force" Evaluation

In a first step, we ran experiments with a small number of sketches to identify the most useful settings. Initially, we tested a large number of combinations of parameters because multiple options for invariances could be enabled or disabled independently and because parameters like the β value for edge detection, the number of angular and radial partitions for ARP, the reduced size of image for IDM and the size of warp range and local context can have arbitrary values. For the latter, we initially had to pick reasonable ranges.

We used a server with two Intel Xeon E5520 QuadCore 2.26 GHz processors and a total of 12 GB of RAM to perform searches simply with all combinations of parameters for some sketches. By ranking the best-performing settings, we were able to reduce the number of combinations for the subsequent evaluation significantly. For β, we selected the values $\{2, 5, 8, 10, 12, 15, 20, 25\}$ for subsequent evaluations as values above 25 did not perform well on any of the sketches. The reason for this is that too many details from the real images would have been removed during edge detection. Smaller steps would be expected not to generate significantly different results as the results of neighboring β values were already very close, as can be seen in the examples presented in Figures 15 to 18 on pages 36-39. For ARP, we selected the combinations (8,4), (8,8), (16,8) for angular and radial partitions. For IDM, we decided to stick to a single resolution of at most 32 pixels on the longer side of the images, a maximum warp range of 5, and a maximum local context of 3. However, we did not consider the option to ignore empty areas for local contexts less than 2.

6.4 Retrieval Quality

The objective of the second step was to assess the ability to retrieve a known item from a collection. Hence, the most important measure for us was the rank at which the known item was found. In order to be a useful tool, the rank has to be significantly smaller than one would expect for browsing the collection in alphabetic or random order and also less than the number of items the user is willing to browse before giving up. The user would expect to find the known item on average after having seen half of the collection – so in this case for 25,000 objects in the collection, this would be after 12,500 if no text search or other filtering based on meta-data was used and this would certainly be more than any user would be willing to browse.

As we expect the QbS system to be used in an interactive way, we would usually expect the user to set the number of desired results (in other words, the number of items the user is willing to browse) when submitting a search and refining the search to her needs, e.g., enabling or disabling some invariances, and re-submitting the search rather than submitting a single search and then browsing until the item was found. However, for the purpose of evaluating the ranks, we always continued to compute results until the known item was found. To remain comparable with what the user would see after some refinements, we performed this search with a smaller number of combinations selected for Section 6.3 and selected the best-performing setting. Figure 11 shows the ranks split up by known item. The ends of the bar show the rank achieved by the best and the worst sketch for this image, the thick grey bar starts at the first quartile and ends at the third quartile and therefore indicates the range of ranks which half of all sketches achieved. Finally, the blue diamond indicates the mean or –in other words– the average rank of all sketches for this image. Notice that the rank axis is in logarithmic scale. Also take into account that for image im18797.jpg in both ARP and IDM, the first and third quartile are both positioned at rank 1, since QbS retrieved the top rank for all of the sketches with a couple of exceptions that gave relatively poor results, which in return increased the value of the mean.

The results show that only a single sketch had a rank slightly worse than 1,000 while the vast majority of sketches achieved a rank below 100, which could already be a number of results a user might be willing to browse. For searching with IDM, even the majority is below 10, a number of results that can easily fit on a single result screen and out of which the user would recognize the image almost instantaneously.

With respect to the parameters, for all searches the choice of the appropriate β value depends on both the known item and the user sketch, with the latter being the one with higher impact. Therefore, it is not possible to come up with a single value of β for a diverse collection like the MIRFLICKR benchmark that would suit all users. However, the QbS system defines a set of values with reasonable limits and lets the user not only pick a single value, but also a range from this set that she can refine and shift during the search process. Figures 15, 16, 17, and 18 show the generated edge maps at different β values for the four images used throughout the evaluation. Furthermore, to visually show how β is dependent on both the known item and the user sketch, we have clustered the user sketches for each known item according to the β that resulted in the best results. To calculate this, we obtained the top 30 configurations that gave the best ranks for each sketch and picked the most frequently occuring β value. The classifications that are done using the ARP

algorithm are shown in Figures 19, 20, 21, and 22 and the classification that are done using the IDM algorithm are shown in Figures 23, 24, 25, and 26. General observations show that the more detailed the sketch is, the lower the values of the threshold β are that give the best rankings since they preserve more edge information. For example, Figures 22(b) and 22(d) are sketches that pertain a high degree of detail, including background information in the form of clouds and therefore β values of 2 and 5 gave the best results for these two sketches. On the other hand, very rough sketches that contain little details tend to give best results with higher values of β. An example of this is evident in Figures 22(m), 22(n), and 22(o) that gave their best results with a higher β value of 15. It is also evident from the classification of sketches according to best performing β in both ARP and IDM that IDM is less sensitive to the β values as it resulted in a smaller number of clusters than in ARP. This can be attributed to the fact that in IDM, local context tends to ignore much of the "clutter" of details in the image.

For ARP, rotation invariance was most helpful for finding im1660.jpg (cf. Figure 7(a); in 4 out of 15 sketches, this improved the ranking). Half of the sketches for im10853.jpg (cf. Figure 8(a); 7 out of 14) improved by ignoring background / completely empty areas. Retrieval results for a majority of sketches (8 out of 15) for im18707.jpg (cf. Figure 9(a)) did improve by enabling scale invariance. Scale invariance was helpful for 3 sketches of im1660.jpg (cf. Figures 7(a)) and im18797.jpg (cf. 10(a)). This shows that all invariances were needed in some cases — however, as one can expect, this heavily depends on the user's sketches. As an example, Figures 12, 13, and 14 show the effect of using an appropriate invariance and a keyword filter for a given sketch. Figure 12 shows the result obtained by QbS when using the given sketch without neither turning on any invariances nor providing any keyword filtering, which results in finding the known item only at position 83. When turning on translation invariance, the result significantly improves and a rank of 7 is achieved as shown in Figure 13. Furthermore, filtering the results with a proper keyword (in this case the keyword "flying"), assists in retrieving the known item at the first position as shown in Figure 14.

For IDM, it is even harder to identify trends as the results are very close. However, there is a tendency that for im1660.jpg (cf. Figures 6(a)) and im18707.jpg (6(c)) a warp range of 5 and a local context of 3 return the best results. This can be explained by the fact that this is the biggest setting for deformation that we allow while still constraining the results through the patch size and therefore being the option in which IDM can cope best with issues in translation and scaling.

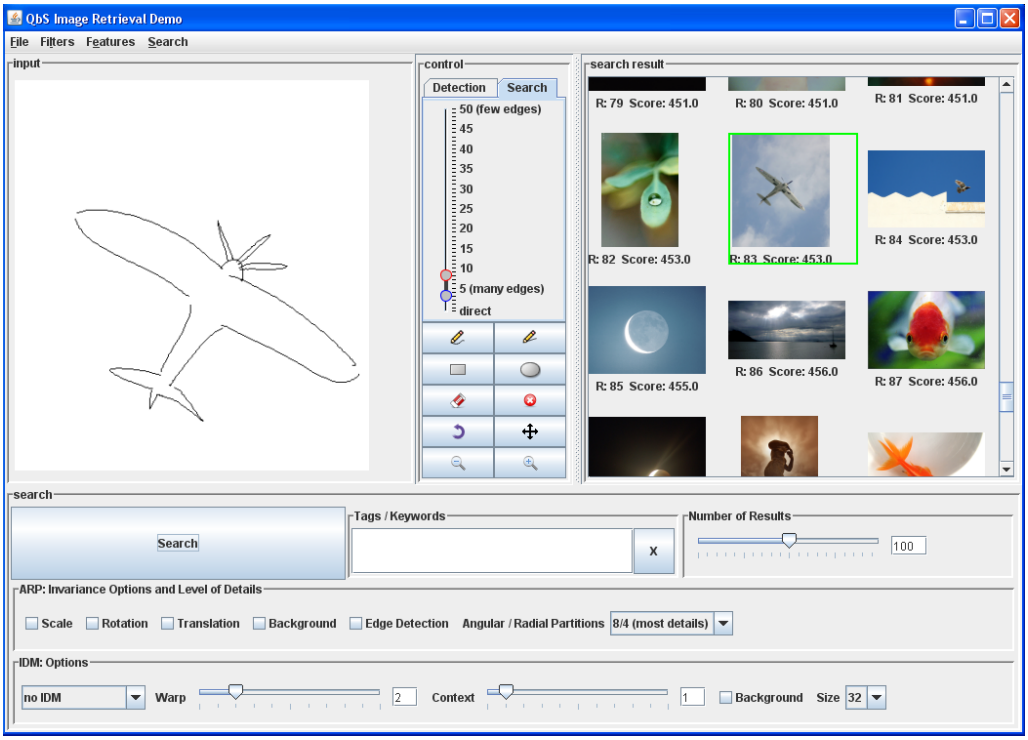


Figure 12: No invariances turned on. Rank achieved: 83

If search via keywords, tags, or other meta-data is performed, the number of images that have to be browsed in worst case shrinks already dramatically – as long as the keywords match the known item. In Table 4, we compare the average ranks achieved by pure content-based retrieval using ARP and IDM with the ranks when combined with keyword search and also with the rank achieved when Lucene returns the item when only the keyword search is used. The column 'Search Space' contains the number of objects that pass the filter, so in case of plain CBIR, all images of the collection and in case of a text filter, the number of hits. Notice that even for words with low selectivity like "sea" the combination with either ARP or IDM results in an average rank below two and in every combination, using the sketch plus a single keyword achieves a better rank than performing a boolean search where two keywords are combined with AND (with a single exception being Paris and using ARP).

This does not mean that keyword-based search does not perform well for this collection. We have selected only terms from a list of tags that people knew and therefore would use for search, even if they did not tag the images themselves. Drawing a sketch and adjusting the search parameters will certainly be a bigger effort than adding another keyword to the search. However, 2,128 out of the 25,000 images (or 8.51%) do not have any tags – even though the MIRFLICKR collection contains only images with highest "interestingness" shared on a very popular website. Less popular or private image collections are expected to have even less tags available. Therefore, there are many situations in which keyword search alone cannot deliver satisfactory results.

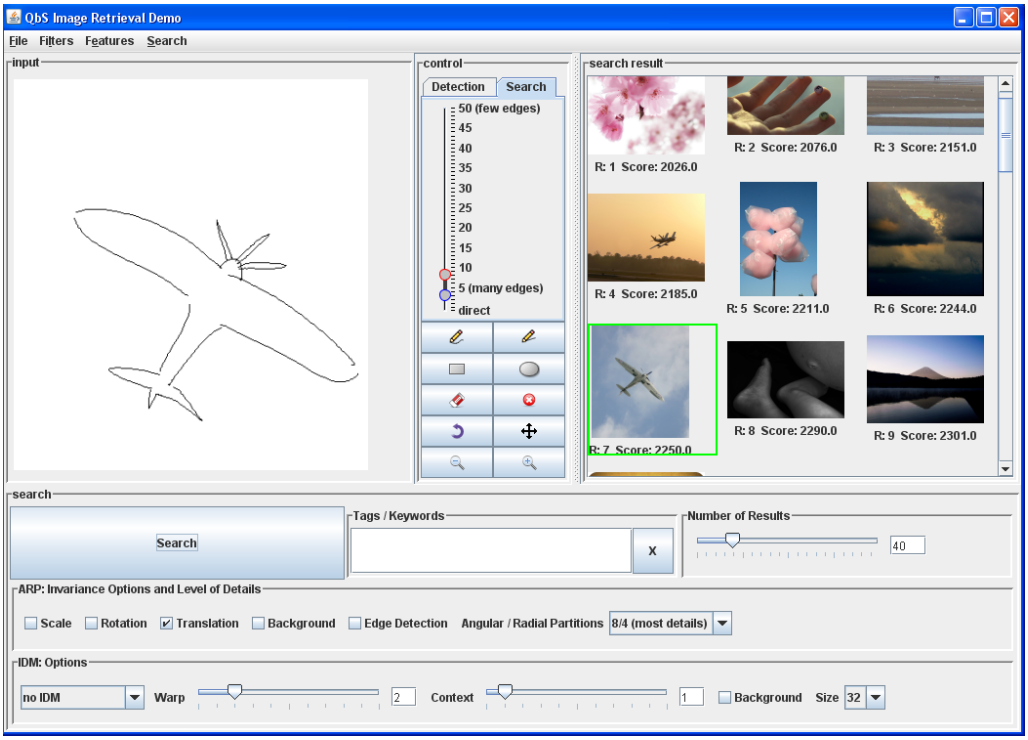


Figure 13: Translation invariance turned on. Rank achieved: 7

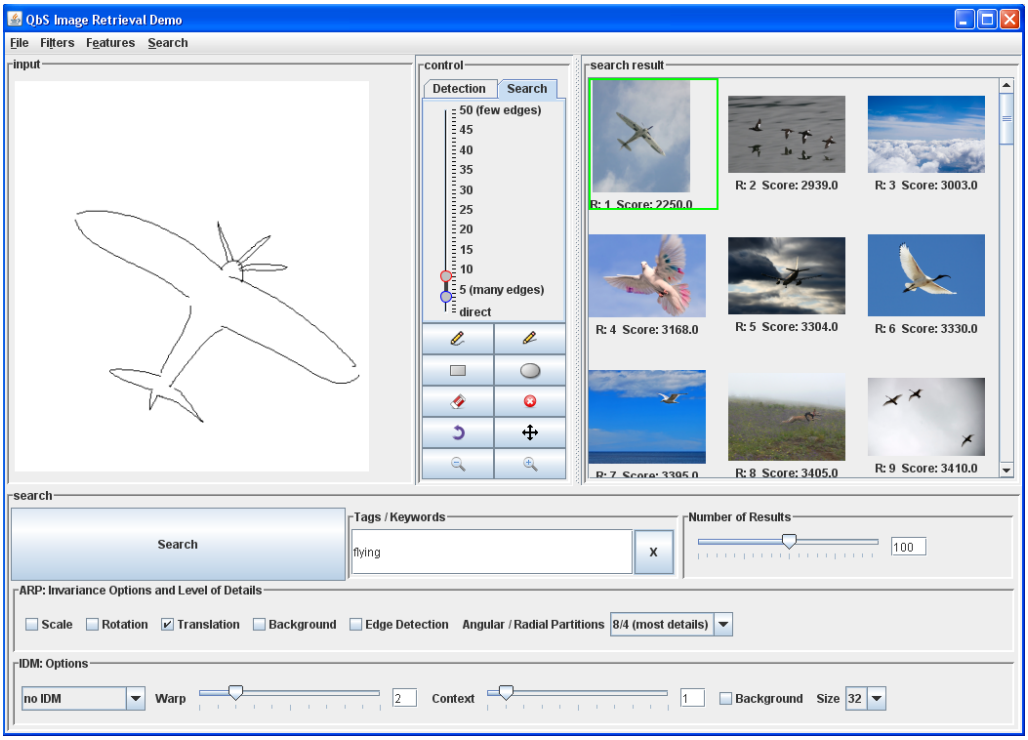


Figure 14: Translation invariance turned on & text filtering enabled with the keyword "flying". Rank achieved: 1

		Text Filter (OFF)		Text Filter (ON)			Search Space	
img	Keywords	ARP	IDM	Lucene	ARP	IDM	Text	CBIR
im1660.jpg	flying	6.33	7.2	56	1	1	57	25,000
	plane			33	1	1	35	
	flying & plane			4	1	1	4	
	flying | plane			6	1	1	88	
im10853.jpg	sea	49.07	15.43	157	1.64	1.43	301	25,000
	mountain			45	1.07	1.07	88	
	sea & mountain			4	1	1	7	
	sea | mountain			4	1.86	1.57	382	
im18707.jpg	bike	118.67	52.4	30	1.13	1.53	111	25,000
	bicycle			21	1.13	1.2	87	
	bike & bicycle			8	1	1.07	37	
	bike | bicycle			8	1.2	1.73	161	
im18797.jpg	paris	61.8	3.3	24	1.2	1	224	25,000
	eiffeltower			1	1	1	13	
	paris & eiffeltower			1	1	1	11	
	paris | eiffeltower			1	1.2	1	226	

Table 4: Average Rank of Known Item

6.5 Retrieval Times

To get realistic results for the retrieval times as the end user will perceive them, all time measurements have been performed on the Lenovo X200t Tablet PC that was also used for the acquisition of the sketches. It has an Intel Core 2 Duo L9400 CPU with 1.86 GHz, 2 GB of RAM, a regular internal 2.5" HDD and is running Windows XP Professional Tablet PC Edition 2005 with Service Pack 3.

Due to the compactness of the ARP features and the early termination strategy, common times for the plain distance computation for up to 200 nearest neighbors are between 15 and 40 milliseconds for a single value of β. It does not exceed 500 milliseconds even for big ranges of β and using many image regions for invariances as long as features are accessed from cache in main memory. As soon as features cannot be read from cache, e.g., directly after the start of the application, the search is I/O bound. Each individual feature file consumes between 3 MB (8/4 partitions) and 13 MB (16/8 partitions) and it takes on average between 0.5 and 2.5 seconds to load it from disk. This time certainly could be optimized, for instance by adopting an appropriate high-dimensional index structure. However, it only is beneficial when additional features have to be read – for subsequent searches, most features can usually be read from the cache and therefore, this only occasionally affects the users' experience during usual query sessions. Also, due to the effect of caching of image thumbnails and loading of additional thumbnails in background, result presentation for subsequent and only slightly modified searches appears to the user usually without noticeable delay.

The features for IDM are considerably larger than for ARP. In uncompressed and simple serialized form, each IDM feature file consumes about 100 MB at a resolution of at most 32 by 32 pixels. For simplicity, we store these files compressed with the GZip algorithm. This reduces disk I/O, but more importantly, saves considerable disk space. The time for performing the search is CPU bound, as minimizing the distance within the warp range is costly, in particular when a large local context has to be considered. So even when performing search with a single value of β and a moderate size with warp range of 2 and a local context of 1, search takes commonly 10 to 15 seconds even when features are cached in memory. This is three orders of magnitude slower than simple ARP searches. For a warp range of 5 and a local context of 3, search may take more than 5 minutes, which is certainly not acceptable for interactive use. Even high-dimensional index structures cannot resolve this issue easily as the image distortion model is not a metric distance measure and uses variable length features with –in our case– a maximum dimensionality of 1024. The easiest "solution" certainly is to switch to a faster machine. The two-processors quad-core Nehalem server that we used for evaluations in Section 6.3 finishes such searches in 20 to 50 seconds. Another, in most cases more appropriate approach, can be to use IDM only together with a filter. It can either be used in combination with a keyword search or in a two-step approach to search and re-rank results generated by ARP. In the case of keywords, even when one of the least selective keywords for this collection is used ("sky" which has 846 hits), and features are read directly from disk, the search takes less than 20 seconds on the Tablet PC for a warp range of 5 and a local context of 3.

7 Conclusions and Outlook

We have presented the QbS approach that assists users in searching for known images on the basis of user-drawn sketches. It incorporates novel input devices and provides support to deal with the inherent challenges that are imposed by comparing user-drawn sketches acquired from such input devices with images in large image collections. In particular, this includes several invariances that allow a user to find the desired result despite of missing background information, misplacements, and/or differences in scale or rotation. Furthermore, the QbS system has been evaluated on the MIRFLICKR-25000 benchmark collection and the results show that query by sketching can in fact help the user to find known images – with or without the use of keywords. Results also show that the retrieval times are sufficiently small to perform interactive searches.

The QbS system can be extended by incorporating color information in addition to the edge information gathered from the sketches, and by evaluating other novel user interfaces, in particular Digital Pens and Interactive Paper. Furthermore, we plan to continue the evaluation of the system with i.) even larger image collections, ii.) different types of image collections, and iii.) more user studies. As larger collection, we plan to evaluate the use of QbS for the extended MIRFLICKR data set with one million images [15]. As one collection with different content, we will apply QbS to the search in image collections of paper watermarks. Such watermarks are extensively used by historians to date ancient paper documents as they are unique enough to commonly identify the period and place in which the paper was produced. In the additional studies, we also want to incorporate additional features, so that we can for instance compare ARP to MPEG-7 feature descriptors.

References

[1] S. Berretti, A. Del Bimbo, and P. Pala. Retrieval by Shape Similarity with Perceptual Distance and Effective Indexing. *IEEE Trans. on Multimedia*, 2(4):225–239, 2000.

[2] K. Bischoff, C. S. Firan, W. Nejdl, and R. Paiu. Can all tags be used for search? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08)*, pages 193–202, 2008.

[3] A. Chalechale, A. Mertins, and G. Naghdy. Edge Image Description using Angular Radial Partitioning. *IEE Proc. on Vision, Image & Signal Processing*, 151(2):93–101, 2004.

[4] A. Chalechale, G. Naghdy, and A. Mertins. Sketch-based image matching using angular partitioning. *IEEE Transactions on Systems, Man and Cybernetics*, 35(1):28–41, 2005.

[5] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2Photo: internet image montage. *ACM Trans. Graph.*, 28(5):1–10, 2009.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR (1)*, pages 886–893. IEEE Computer Society, 2005.

[7] A. Del Bimbo and P. Pala. Visual Image Retrieval by Elastic Matching of User Sketches. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(2):121–132, 1997.

[8] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. A descriptor for large scale image retrieval based on sketched feature lines. In *ACM Symposium on Sketch-Based Interfaces and Modeling*, Aug. 2009.

[9] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. PhotoSketch: A Sketch based Image Query and Compositing System. In *SIGGRAPH*, 2009.

[10] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. An evaluation of descriptors for large-scale image retrieval from sketched feature lines. In *Computers and Graphics Journal (to appear)*, 2010.

[11] M. Flickner et al. Query by Image and Video Content: The QBIC System. *Computer*, 28(9):23–32, 1995.

[12] F. Guimbretière. Paper augmented digital documents. In *Proc. of ACM symposium on User interface software and technology (UIST '03)*, pages 51–60, New York, NY, USA, 2003.

[13] K. Hirata and T. Kato. Query by visual example. *Advances in Database Technology EDBT*, 92:56–71, 1992.

[14] M. J. Huiskes and M. S. Lew. The MIR Flickr Retrieval Evaluation. In *Proceedings of the ACM International Conference on Multimedia Information Retrieval (MIR'08)*, 2008.

[15] M. J. Huiskes, B. Thomee, and M. S. Lew. New Trends and Ideas in Visual Concept Detection: The MIR Flickr Retrieval Evaluation Initiative. In *Proceedings of the ACM International Conference on Multimedia Information Retrieval (MIR '10)*, pages 527–536, New York, NY, USA, 2010. ACM.

[16] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. In *Proc. of SIGGRAPH '95*, pages 277–286, 1995.

[17] D. Keysers, T. Deselaers, C. Gollan, and H. Ney. Deformation models for image recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(8):1422–1435, 2007.

[18] D. Keysers, C. Gollan, and H. Ney. Local context in non-linear deformation models for handwritten character recognition. In *Proc. of ICPR'04*, volume 4, pages 511–514, Washington, DC, USA, 2004.

[19] B. S. Manjunath, J. rainer Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Trans. on Circuits and Systems for Video Technology*, 11:703–715, 2001.

[20] S. McDonald and J. Tait. Search strategies in content-based image retrieval. In *Proc. of ACM SIGIR '03*, pages 80–87, 2003.

[21] W. Niblack et al. The QBIC Project: querying images by content, using color, texture, and shape. In W. Niblack, editor, *Proc. of SPIE Vol. 1908*, pages 173–187, Apr. 1993.

[22] M. C. Norrie, B. Signer, and N. Weibel. General Framework for the Rapid Development of Interactive Paper Applications. In *Proc. CoPADD'06*, pages 9–12, 2006.

[23] S. J. Park, D. K. Park, and C. S. Won. Core experiments on MPEG-7 edge histogram descriptor. *MPEG document M5984*, May 2000.

[24] T. Pavlidis. Limitations of content-based image retrieval. invited plenary talk at the 19th International Conference on Pattern Recognition (ICPR 2008), Dec. 2008. Slides can be found in `http://www.theopavlidis.com/technology/CBIR/PaperB/icpr08.htm`, with an an extended version at `http://www.theopavlidis.com/technology/CBIR/PaperB/vers3.htm`.

[25] H. Reiterer and T. Büring. Zooming techniques. In L. Liu and T. M. Özsu, editors, *Encyclopedia of Database Systems*, pages 3684–3689. Springer, Nov. 2009.

[26] S. Romdhani, J. Ho, T. Vetter, and D. J. Kriegman. Face Recognition Using 3-D Models: Pose and Illumination. *Proc. of the IEEE*, 94(11), Nov. 2006.

[27] G. Salton and M. J. Mcgill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[28] T. Sikora. The MPEG-7 visual standard for content description-an overview. *IEEE Trans. Circuits Syst. Video Techn.*, 11(6):696–702, 2001.

[29] M. Springmann, A. Dander, and H. Schuldt. Improving efficiency and effectiveness of the image distortion model. *Pattern Recognition Letters*, 29(15):2018–2024, 2008.

[30] M. Springmann, A. Ispas, H. Schuldt, M. C. Norrie, and B. Signer. Towards Query by Sketch. In *Proceedings of the Second International DELOS Conference*, Dec. 2007.

[31] M. Springmann and H. Schuldt. Using Regions of Interest for Adaptive Image Retrieval. In *Proceedings of the Second International Workshop on Adaptive Information Retrieval (AIR 2008)*, Oct. 2008.

[32] R. Weber, H.-J. Schek, and S. Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *Proceedings of 24rd International Conference on Very Large Data Bases (VLDB'98)*, pages 194–205. Morgan Kaufmann, 1998.

[33] S. Westman, A. Lustila, and P. Oittinen. Search strategies in multimodal image retrieval. In *Proceedings of International Symposium on Information Interaction in Context (IIiX '08)*, pages 13–20, 2008.

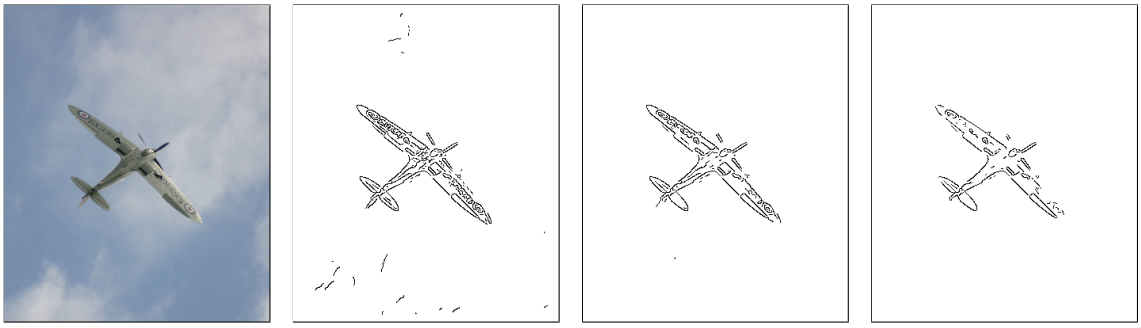
[34] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Proc. of ICCV 2005*, volume 1, 2005.

[35] B. Wu and R. Nevatia. Simultaneous Object Detection and Segmentation by Boosting Local Shape Feature based Classifier. *Proc. of CVPR '07)*, pages 1–8, June 2007.

[36] H. I. Xie. Planned and Situated Aspects in Interactive IR: Patterns of User Interactive Intentions and Information Seeking Strategies. In *Proc. ASIS*, pages 101–110, 1997.

Appendices

A Generated Edge Maps for Known Items

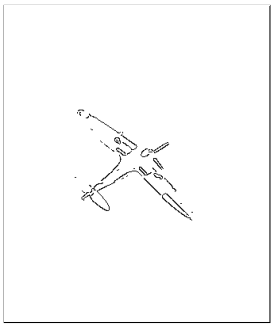


(a)

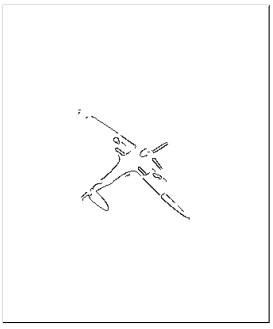
(b) $\beta = 2$

(c) $\beta = 5$

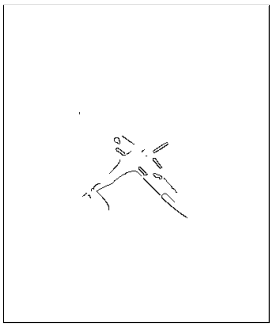
(d) $\beta = 8$



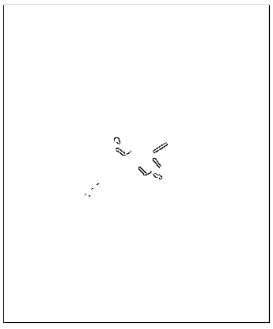
(e) $\beta = 10$



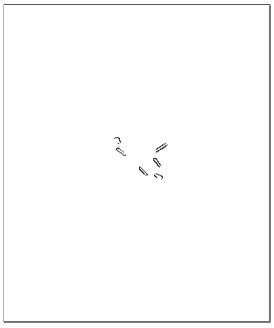
(f) $\beta = 12$



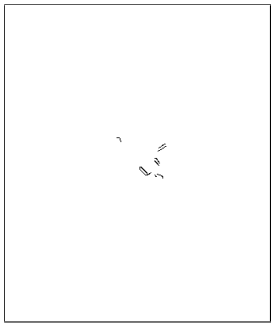
(g) $\beta = 15$



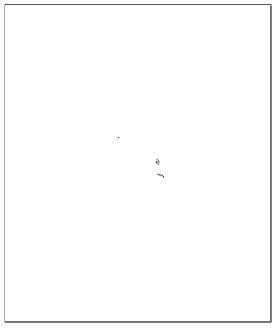
(h) $\beta = 20$



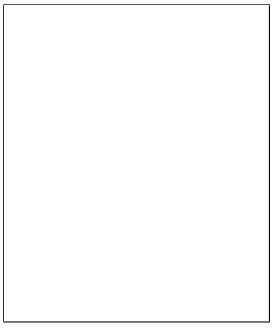
(i) $\beta = 25$



(j) $\beta = 30$

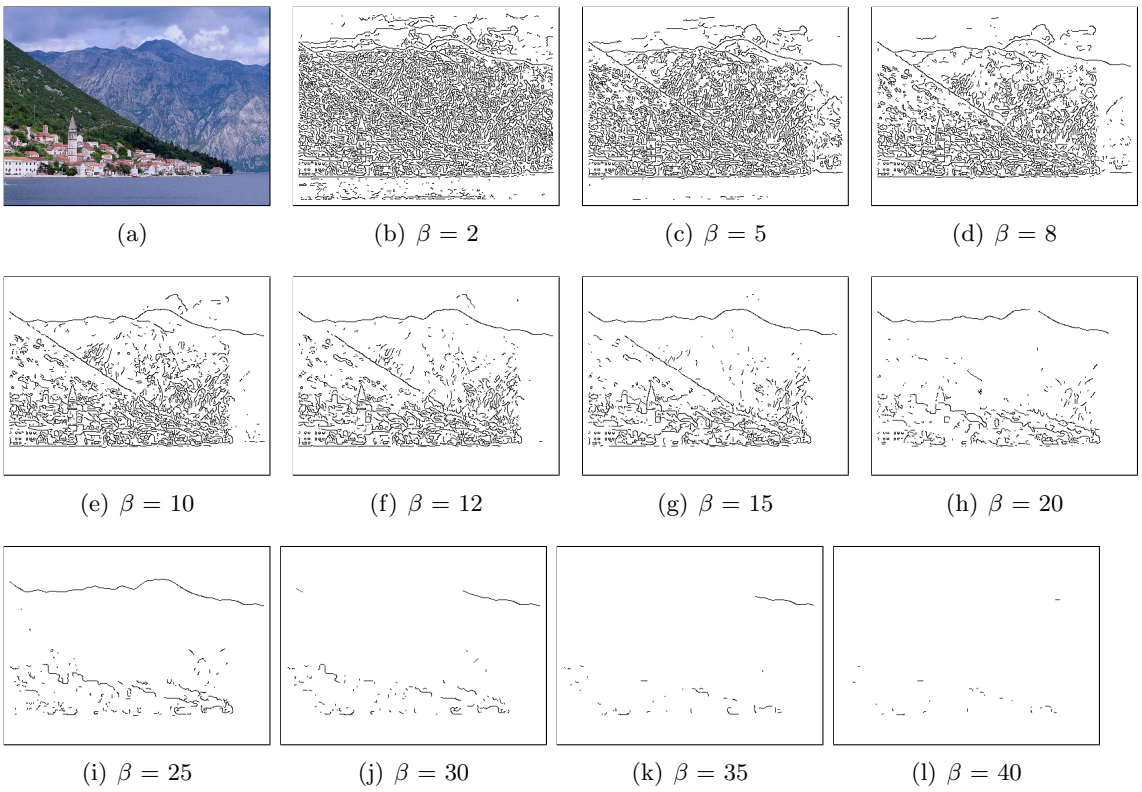


(k) $\beta = 35$



(l) $\beta = 40$

Figure 15: Edge maps for image im1660.jpg at different β values

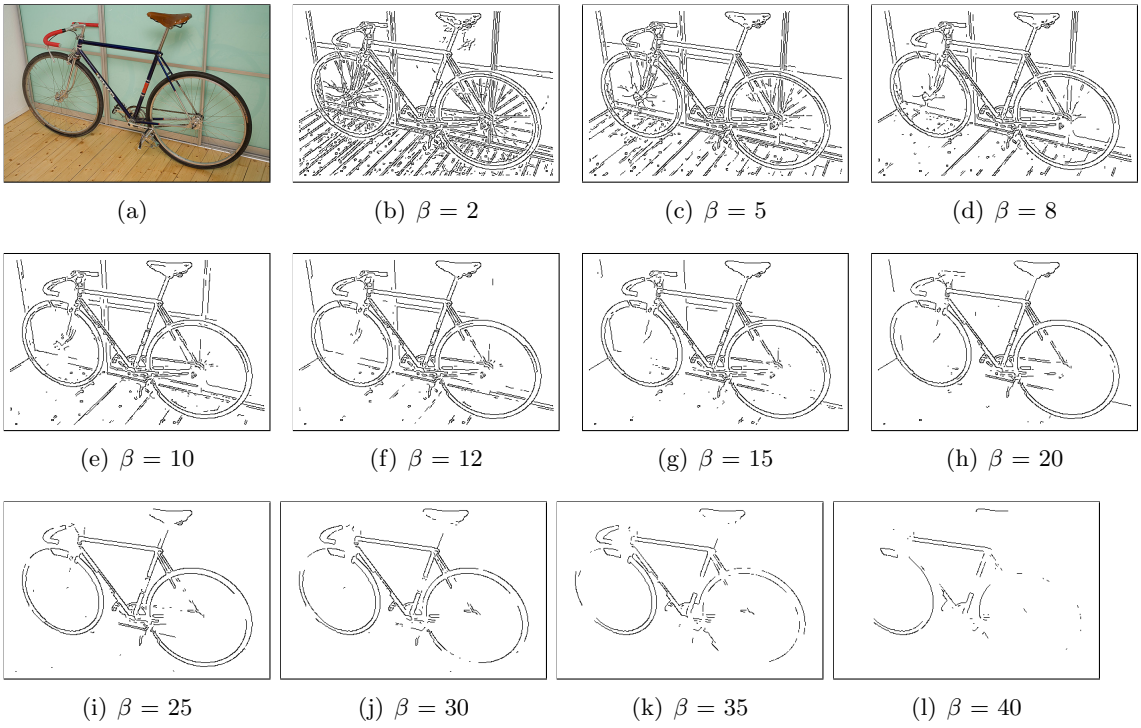


(a) (b) $\beta = 2$ (c) $\beta = 5$ (d) $\beta = 8$

(e) $\beta = 10$ (f) $\beta = 12$ (g) $\beta = 15$ (h) $\beta = 20$

(i) $\beta = 25$ (j) $\beta = 30$ (k) $\beta = 35$ (l) $\beta = 40$

Figure 16: Edge maps for image im10853.jpg at different β values

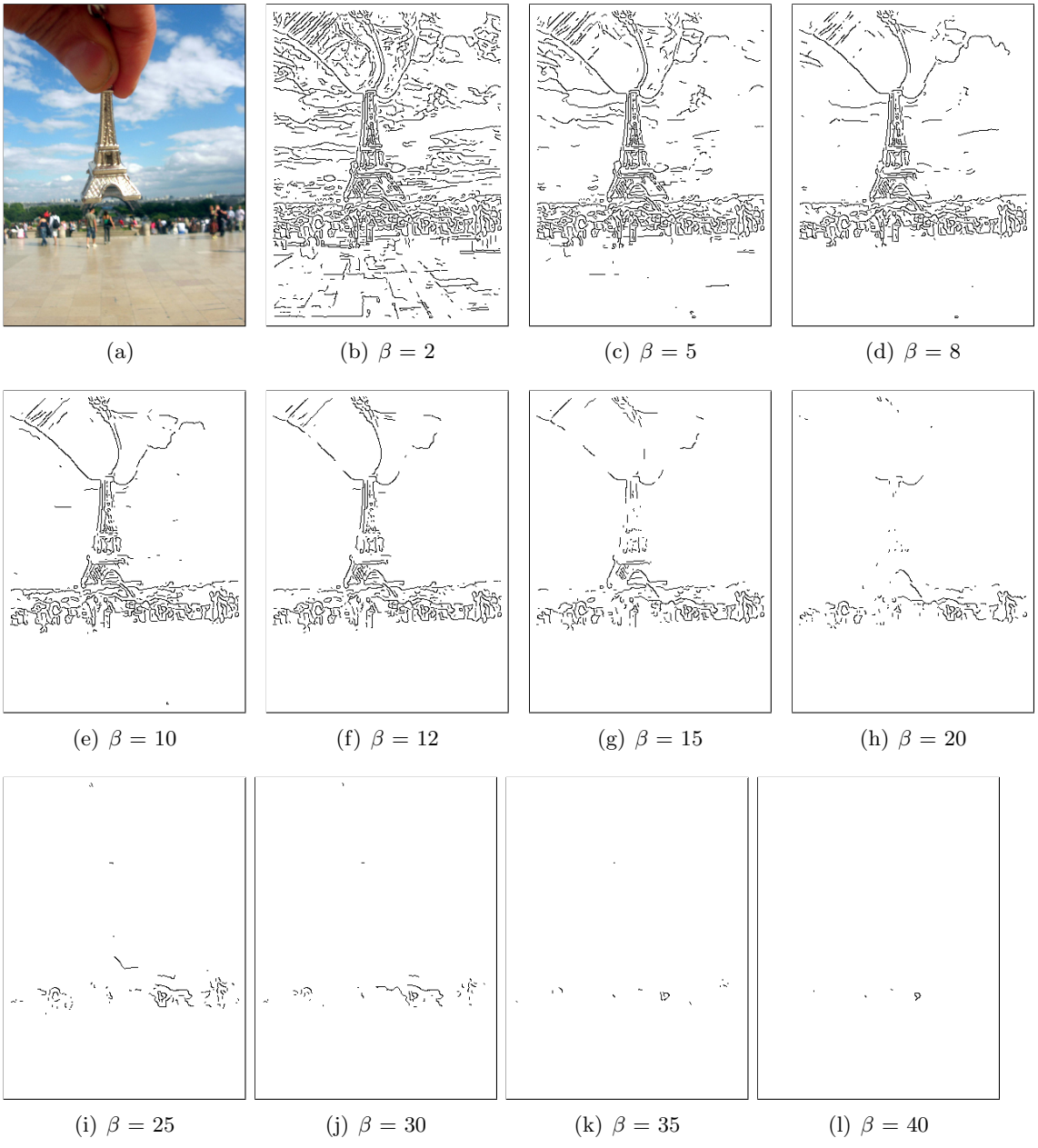


(a) (b) $\beta = 2$ (c) $\beta = 5$ (d) $\beta = 8$

(e) $\beta = 10$ (f) $\beta = 12$ (g) $\beta = 15$ (h) $\beta = 20$

(i) $\beta = 25$ (j) $\beta = 30$ (k) $\beta = 35$ (l) $\beta = 40$

Figure 17: Edge maps for image im18707.jpg at different β values



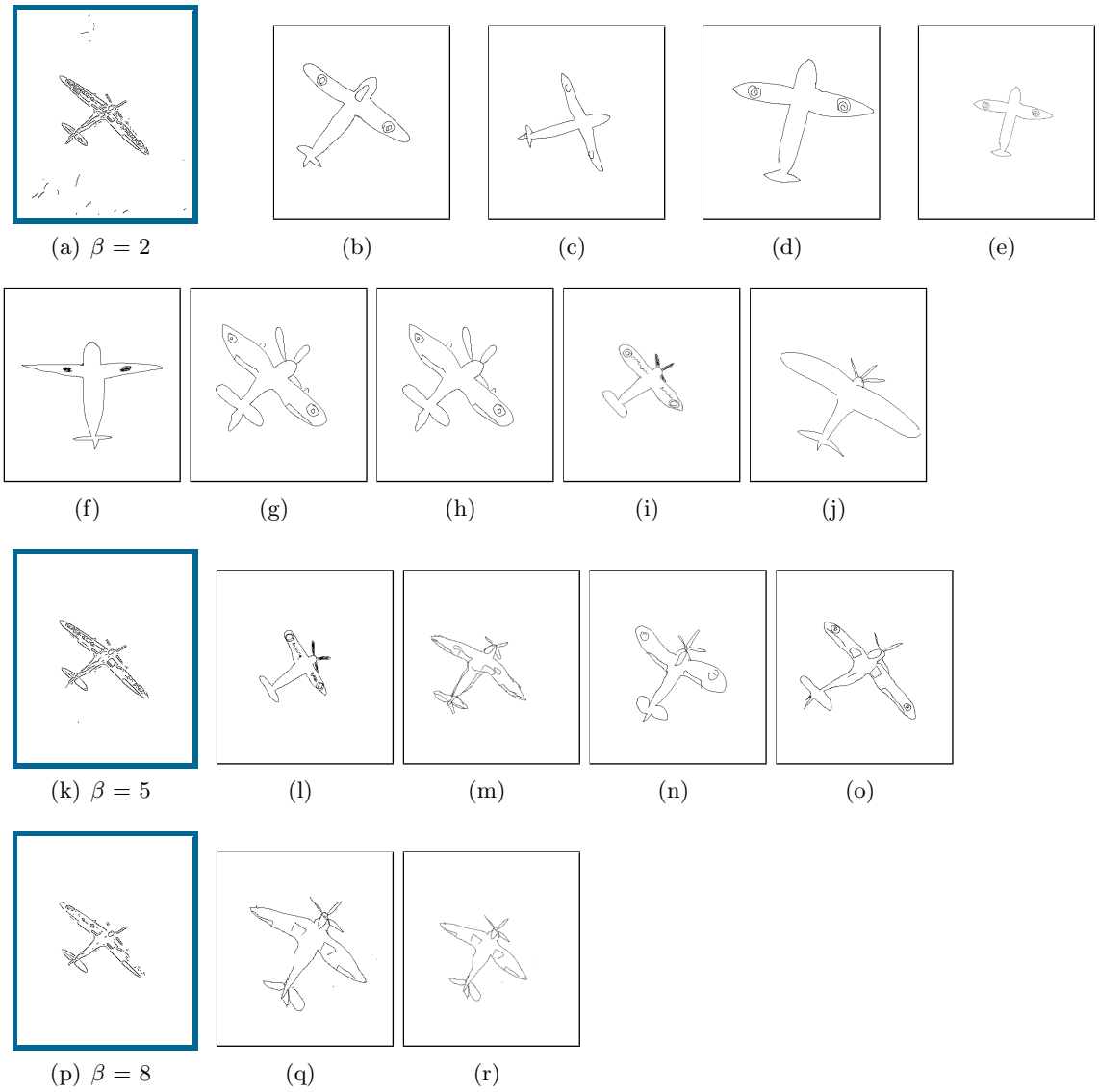
(a) (b) $\beta = 2$ (c) $\beta = 5$ (d) $\beta = 8$

(e) $\beta = 10$ (f) $\beta = 12$ (g) $\beta = 15$ (h) $\beta = 20$

(i) $\beta = 25$ (j) $\beta = 30$ (k) $\beta = 35$ (l) $\beta = 40$

Figure 18: Edge maps for image im18797.jpg at different β values

B User Drawn Sketches Clustered with the Edge Map with the Most Fitting β using ARP



(a) $\beta = 2$ (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) $\beta = 5$ (l) (m) (n) (o)

(p) $\beta = 8$ (q) (r)

Figure 19: Sketches of im1660.jpg clustered by the edge map with the most fitting β

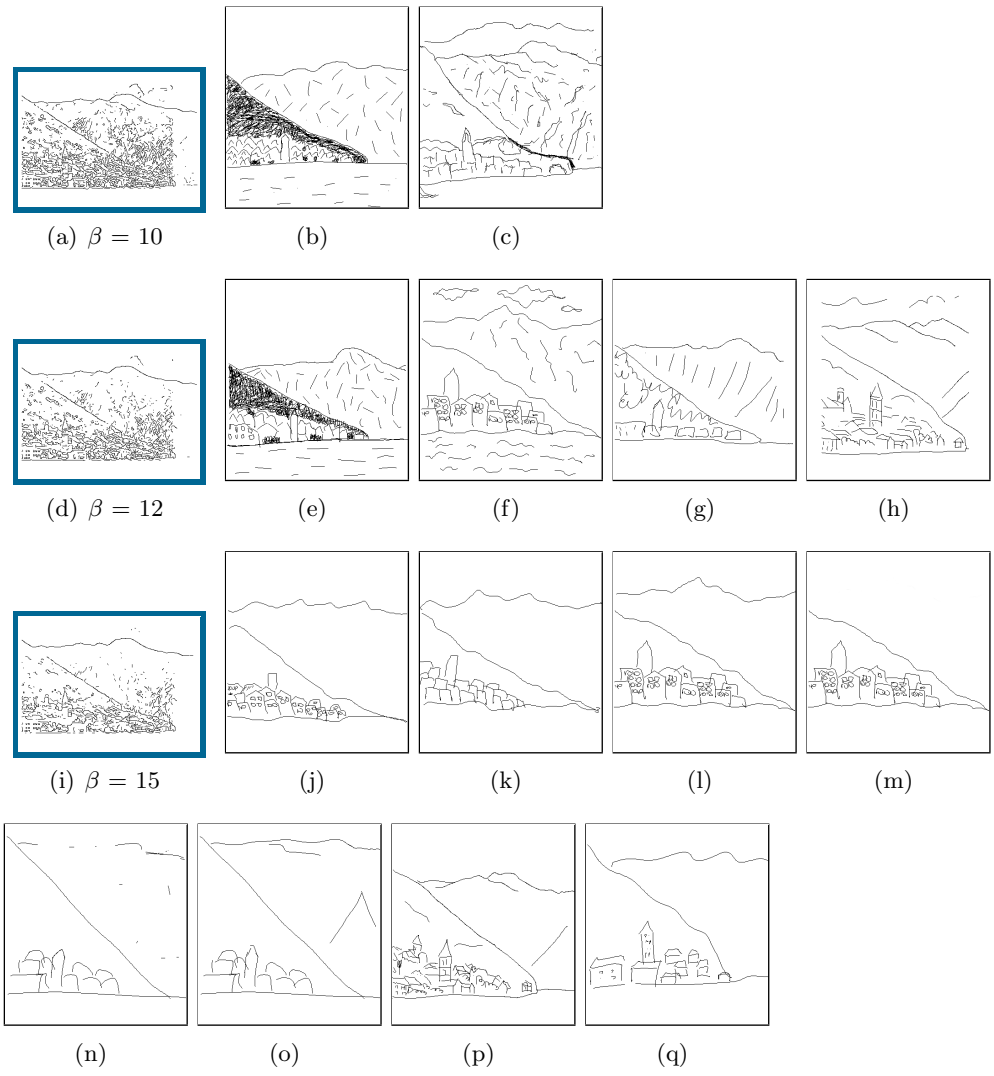
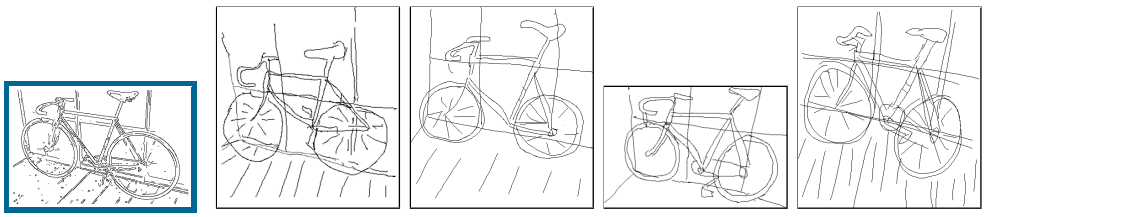
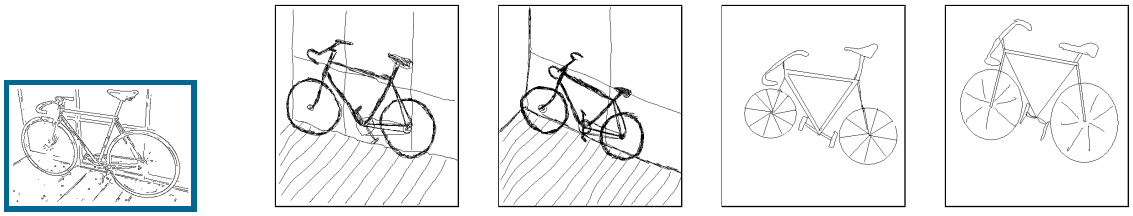


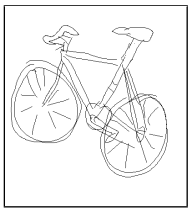
Figure 20: Sketches of im10853.jpg clustered by the edge map with the most fitting β



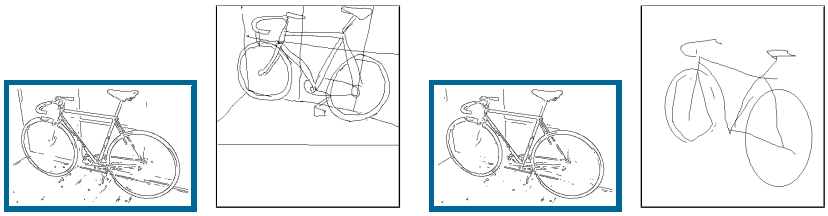
(a) $\beta = 8$ (b) (c) (d) (e)



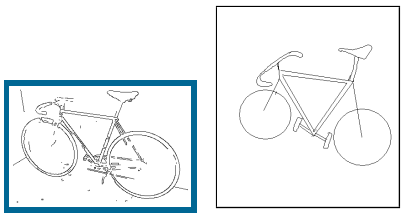
(f) $\beta = 10$ (g) (h) (i) (j)



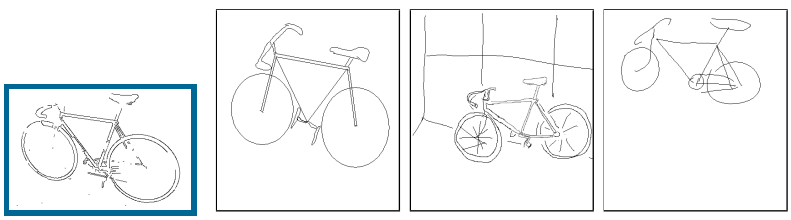
(k)



(l) $\beta = 12$ (m) (n) $\beta = 15$ (o)

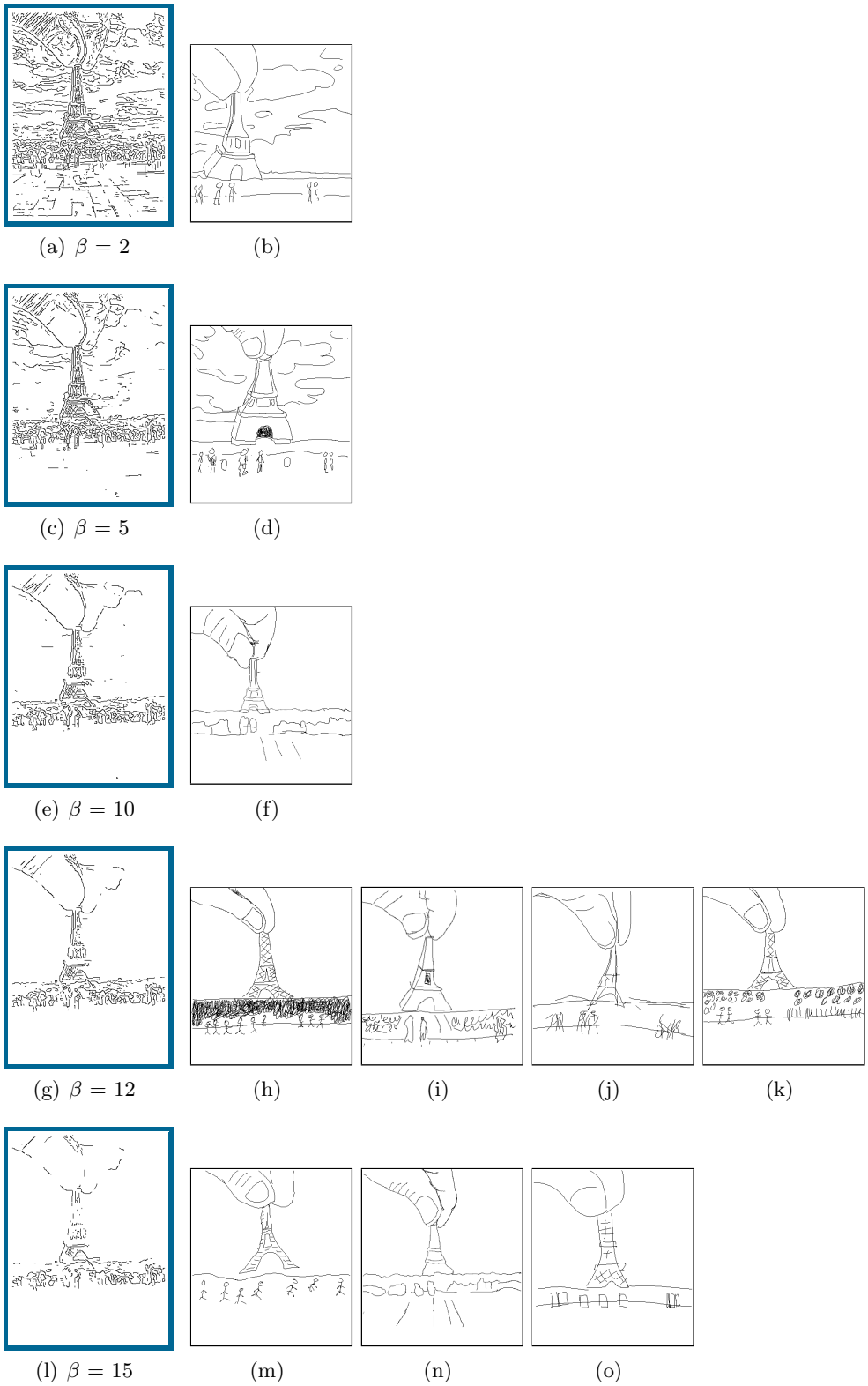


(p) $\beta = 20$ (q)



(r) $\beta = 25$ (s) (t) (u)

Figure 21: Sketches of im18707.jpg clustered by the edge map with the most fitting β



(a) $\beta = 2$ (b)

(c) $\beta = 5$ (d)

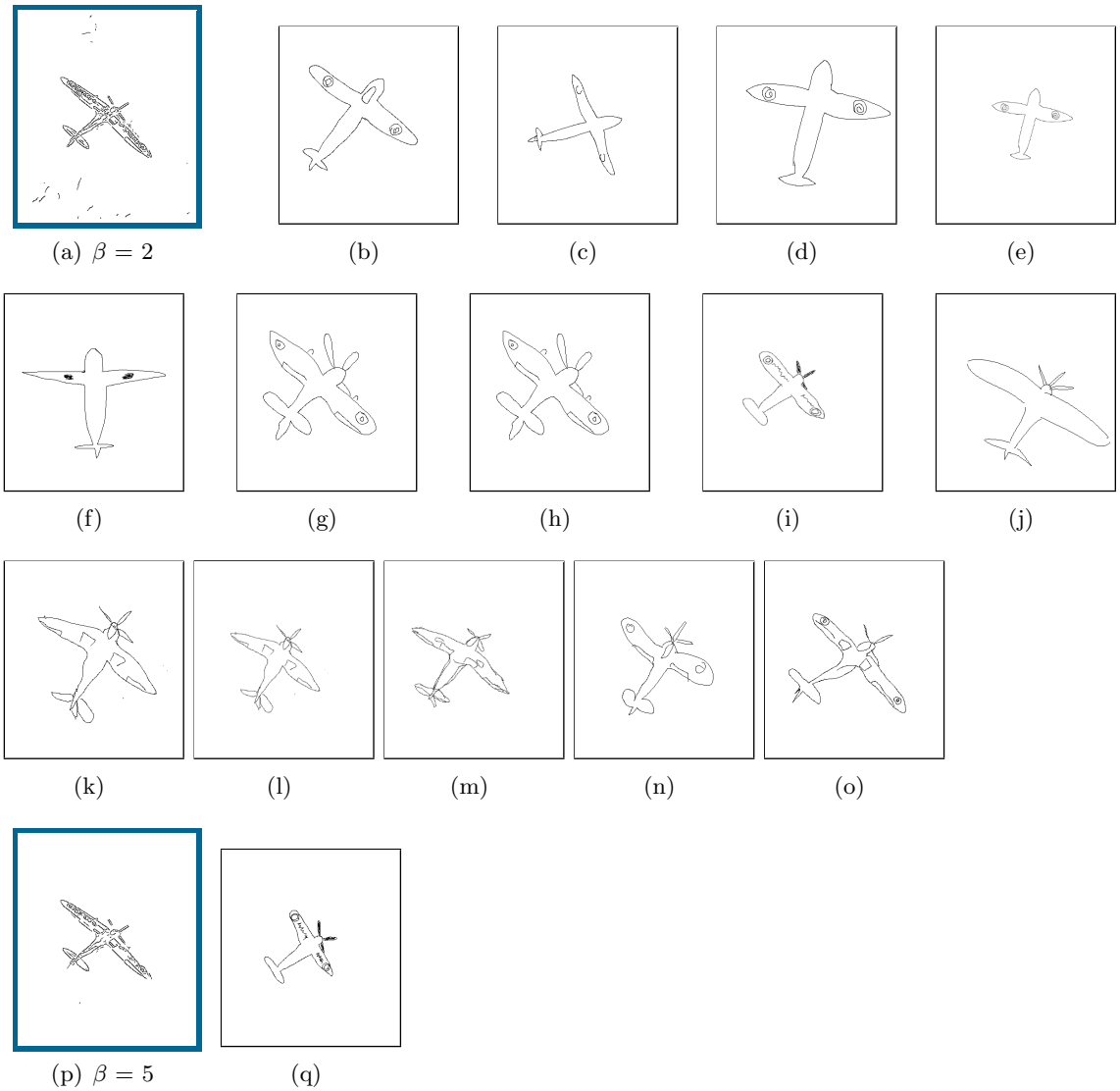
(e) $\beta = 10$ (f)

(g) $\beta = 12$ (h) (i) (j) (k)

(l) $\beta = 15$ (m) (n) (o)

Figure 22: Sketches of im18797.jpg clustered by edge map with the most fitting β

C User Drawn Sketches Clustered with the Edge Map with the Most Fitting β Using IDM



(a) $\beta = 2$ (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

(p) $\beta = 5$ (q)

Figure 23: Sketches of im1660.jpg clustered by the edge map with the most fitting β

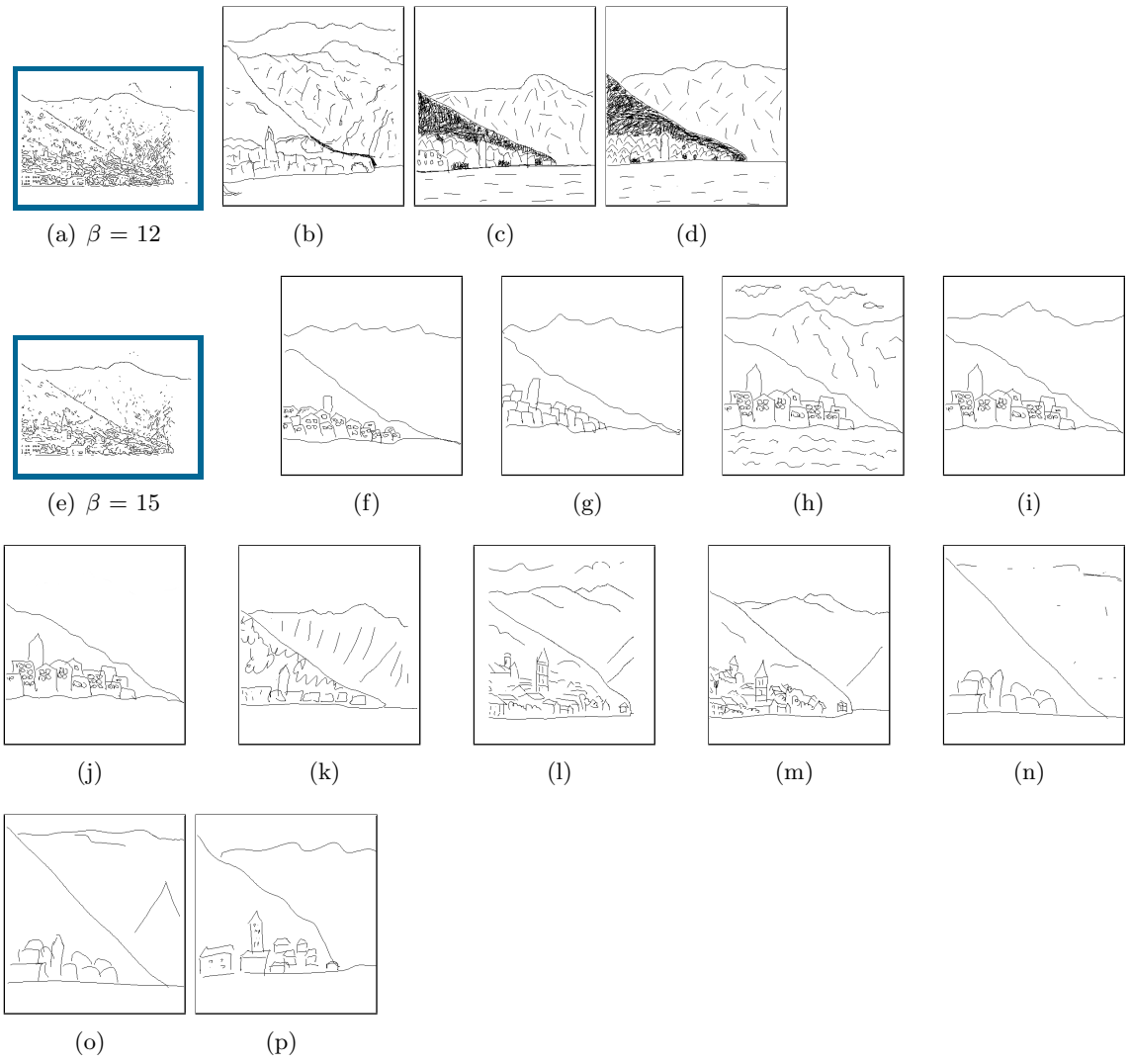
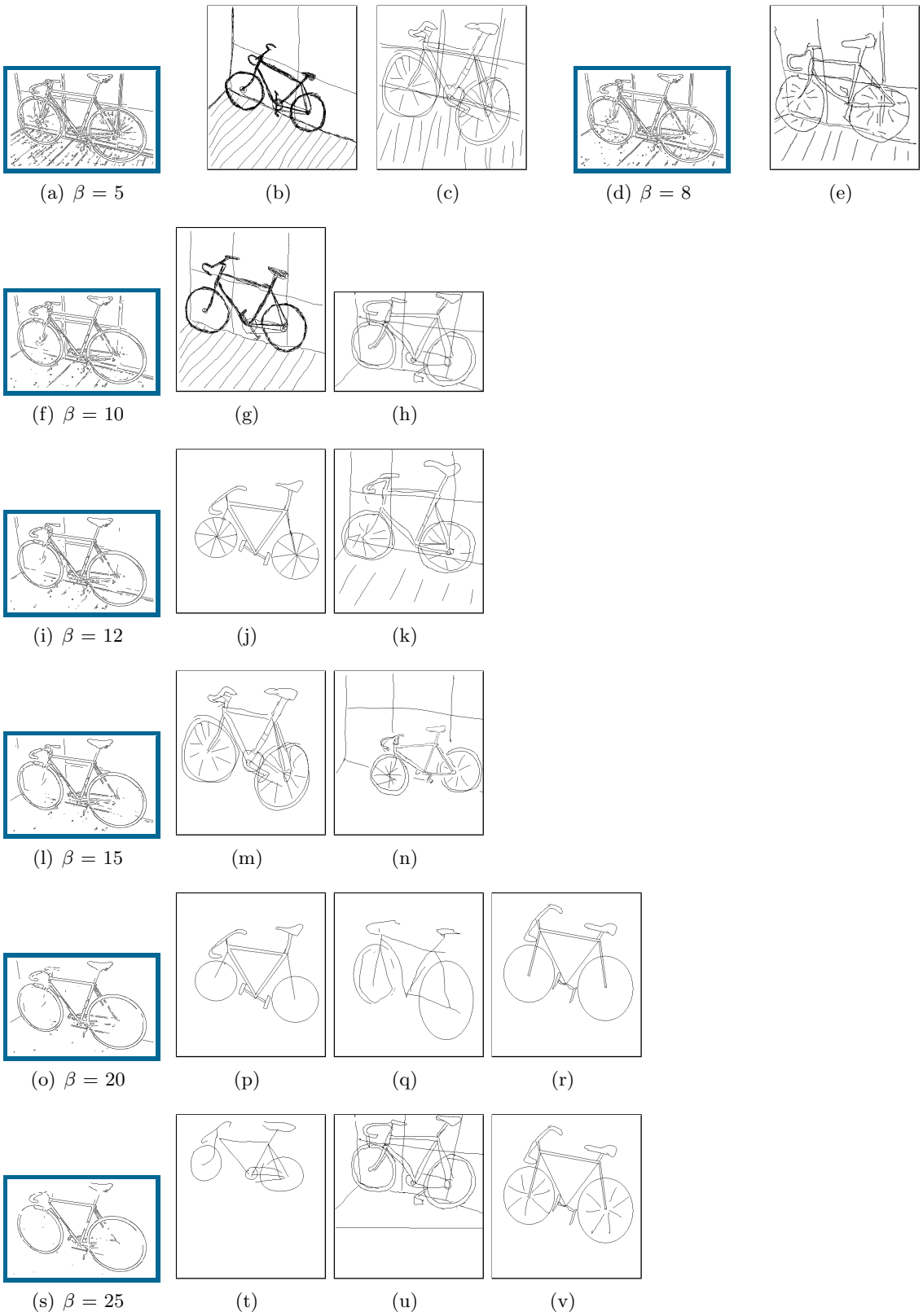


Figure 24: Sketches of im10853.jpg clustered by the edge map with the most fitting β



(a) $\beta = 5$ (b) (c) (d) $\beta = 8$ (e)

(f) $\beta = 10$ (g) (h)

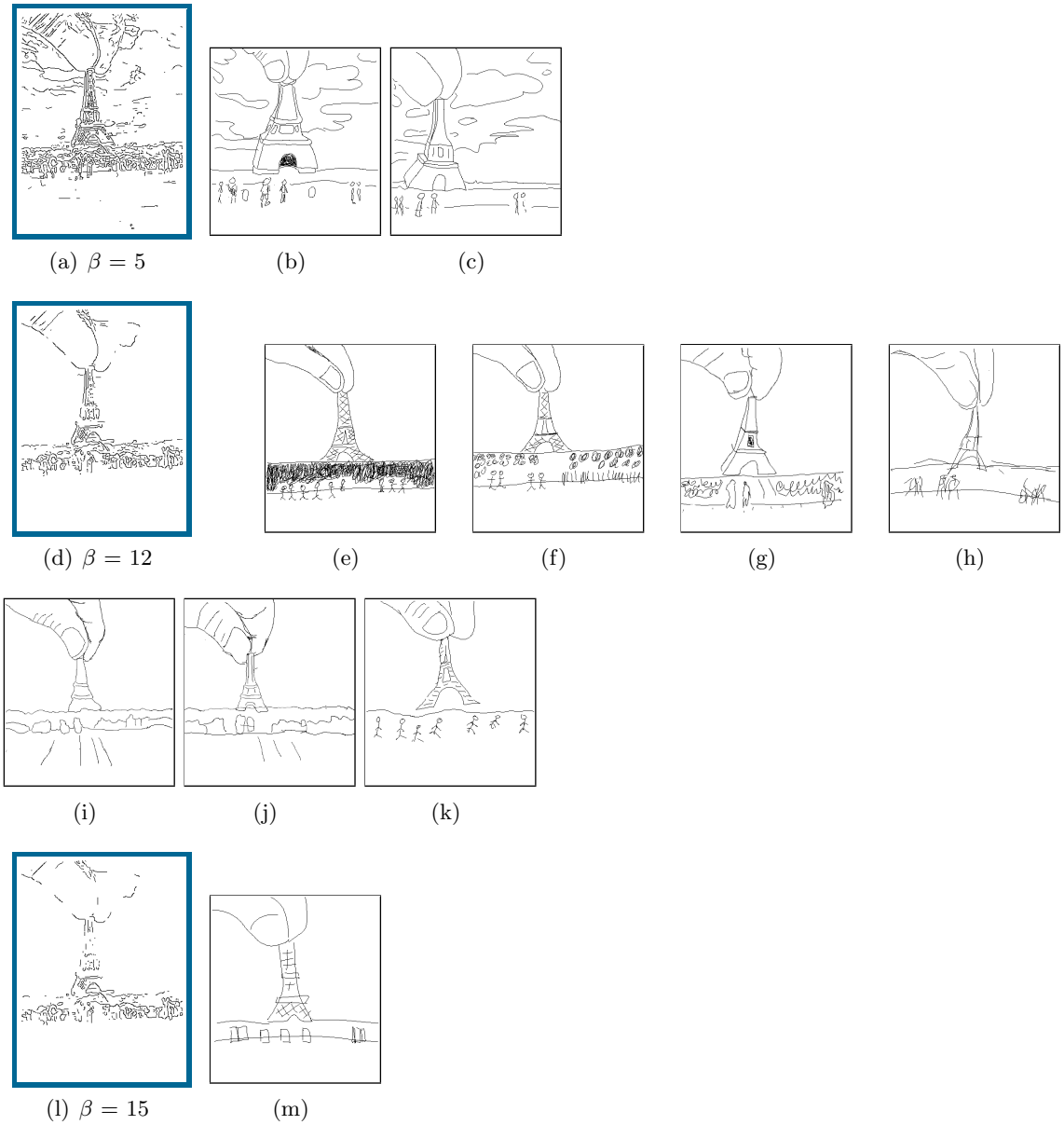
(i) $\beta = 12$ (j) (k)

(l) $\beta = 15$ (m) (n)

(o) $\beta = 20$ (p) (q) (r)

(s) $\beta = 25$ (t) (u) (v)

Figure 25: Sketches of im18707.jpg clustered by the edge map with the most fitting β



(a) $\beta = 5$ (b) (c)

(d) $\beta = 12$ (e) (f) (g) (h)

(i) (j) (k)

(l) $\beta = 15$ (m)

Figure 26: Sketches of im18797.jpg clustered by edge map with the most fitting β