MULTIMEDIA TOOLS AND APPLICATIONS manuscript No.
(will be inserted by the editor)

Mobile Multi-View Object Image Search

Fatih Çalışır · Muhammet Başţan · Özgür Ulusoy · Uğur Güdükbay

Received: date / Accepted: June 2016

Abstract High user interaction capability of mobile devices can help improve the accuracy of mobile visual search systems. At query time, it is possible to capture multiple views of an object from different viewing angles and at different scales with the mobile device camera to obtain richer information about the object compared to a single view and hence return more accurate results. Motivated by this, we propose a new multi-view visual query model on multi-view object image databases for mobile visual search. Multi-view images of objects acquired by the mobile clients are processed and local features are sent to a server, which combines the query image representations with early/late fusion methods and returns the query results. We performed a comprehensive analysis of early and late fusion approaches using various similarity functions, on an existing single view and a new multi-view object image database. The experimental results show that multi-view search provides significantly better retrieval accuracy compared to traditional single view search.

Keywords Mobile visual search · multi-view search · bag of visual words · fusion

1 Introduction

Smart mobile devices have become ubiquitous. They are changing the way people access information. They have some advantages and disadvantages, com-

F. Çalışır, Ö. Ulusoy, U. Güdükbay
Bilkent University, Department of Computer Engineering, Bilkent 06800 Ankara, Turkey
Tel.: +90-312-2901386; Fax: +90-312-2664047
E-mail: fatih.calisir@bilkent.edu.tr, oulusoy@cs.bilkent.edu.tr, gudukbay@cs.bilkent.edu.tr

M. Başţan
Turgut Özal University, Department of Computer Engineering, Etlik, Keçiören 06010, Ankara, Turkey
E-mail: mbastan@turgutozal.edu.tr

pared to regular PCs. The advantages are higher accessibility, easier user interaction and the ability to provide context information (e.g., location) using extra sensors, like GPS and compass. The disadvantages are limited computational power, storage, battery life and network bandwidth [31], although these are constantly being improved and will be less of an issue in the future.

One traditional way to access information on a mobile device is via text search, by entering a few keywords as query (query-by-keyword); but this is usually cumbersome and slow, considering the small screen size of mobile devices. As a convenience, it is also possible to initiate text queries via speech, if automatic speech recognition is available. Sometimes, it is very difficult to express a query using only keywords. For instance, when a user at a shoe store wants to know more about a specific type of shoe (e.g., cheaper prices elsewhere, customer reviews), she cannot easily formulate a text query to express her intent. It is much easier to take a photo of the shoe with the mobile device camera, initiate a visual search and retrieve visually similar results. This is now possible, owing to the recent hardware/software developments in mobile device technology, which turned the smart phones with high-resolution cameras, image processing capabilities and Internet connection into indispensable personal assistants. This in turn triggered research interest in mobile visual search and recognition [8,11,12], and motivated the industry to develop mobile visual search applications, such as Google Goggles [13], CamFind [5], Nokia Point & Find [27], Amazon Flow [1], Kooaba image recognition [29].

The main focus of this work is to leverage the user interaction potential of mobile devices to achieve higher visual search performance, and hence, provide the users with easier access to richer information. One potential application area is *mobile product search*. When the user wants to search for a specific object, she can take a photo of the object to initiate a visual search. Additionally, she can easily tap on the screen to mark the object-of-interest and provide extra information to the search system to suppress the influence of background in matching [39]. More importantly, the user can take multiple photos of the object-of-interest from different viewing angles and/or at different scales, thereby providing much more information about the query object. We refer to *multi-view object image search* as providing multiple query images of an object from various viewing angles and at various scales and combining the query images using early/late fusion strategies to achieve higher retrieval precision on single– and/or multi-view object image databases. High precision on mobile retrieval is especially critical because the screen is too small to display many results, and more importantly, the user usually does not have much time and patience to check more than $10-20$ results.

Multi-view search is different from multi-image search. In multi-image search, multiple images of an object category are used to perform a search [2, 33,36]; the images do not belong to the same instance of the queried object. In multi-view search, query images belong to the same object instance. To illustrate the benefits of multi-view object image search, consider the multi-view images of two different shoes in Figure 1, taken from four different viewing angles at the same scale. Such images are typical on online stores, i.e., multi-

Fig. 1 Multi-view images of two different shoes. Online stores typically contain multi-view images of products.

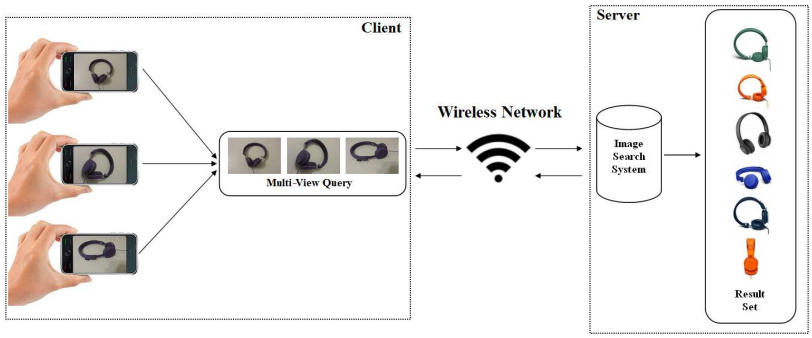

Fig. 2 Client-server architecture of our mobile multi-view visual search system.

view images of objects on clean backgrounds. Assuming the database contains such multi-view images for each object, when a user performs a search using a photo that is close to one of the available views, the results she will get will be better than when the query image has a completely different view. Intuitively, if the user takes multiple photos of the object from different viewing angles, the chance that the query images are similar to the ones in the database will increase. This is also valid when the database contains single view images of each object. The effect of multiple scales is similar. In summary, at query time, the user does not know the view and scale of the object images in the database; by taking multiple photos from different views and at different scales, she can increase the chance of capturing views similar to the database images. This is enabled by the interactivity of the mobile devices; on a non-mobile platform, like a PC, it would be difficult to obtain such multi-view images of an object and perform a multi-view query. Therefore, such a multi-view search system is most suitable for mobile devices with a camera.

In this paper, we address the following questions concerning multi-view object image search:

- Is a multi-view object image database better than a single view database in terms of retrieval precision, and if so how much?
- Do multi-view queries improve retrieval precision on single view or multi-view object image databases, and if so how much?
- Are multi-view queries better than multi-image queries in terms of retrieval precision, and if so how much?

- Multi-view queries need special treatment to combine multiple query/database images using early/late fusion strategies [26,40]. What are the best similarity functions and early/late fusion methods for a search system employing multi-view queries or databases?
- What is the additional computational cost of multi-view search, and is the performance improvement worth the additional cost?

To the best of our knowledge, there is no work describing a multi-view object image search system, addressing these issues. We show through extensive experiments that multi-view queries and/or databases improve retrieval precision significantly, over both single view and multi-image queries, at a cost of modest increase in computation time due to the increase in the number of images to be processed.

To demonstrate the benefits of multi-view search, we built a mobile visual search system based on client-server architecture (cf. Figure 2), using the well-known bag-of-visual-words (BoW) approach [11,12]. We constructed a multi-view object image database and performed extensive experiments on both single and multi-view object image databases with single, multi-image and multi-view queries using various similarity functions and fusion strategies, and presented the results in a systematic way.

2 Related Work

Due to the recent advances in mobile devices with cameras, there has been a growing interest in mobile visual search. Research works investigate different aspects of mobile visual search, such as architectures, power efficiency, speed, and user interaction. Chen and Girod [8] describe a mobile product recognition system where the products are CDs, DVDs and books that have printed labels. The system is local feature based, and Compressed Histogram of Gradients (*CHOG*) and Scale-Invariant Feature Transform (*SIFT*) are used as local features. Two client-server architectures are implemented and compared in terms of response time: one is sending images, the other one is extracting features on the client and sending the features. Sending features took five seconds, sending images took ten seconds to respond. This means that over slow connections like 3G it is faster to extract and send features. Based on this finding, we preferred the former approach in our implementation (Figure 6).

Storage space and retrieval speed are critical in mobile visual search. Girod et al. [12] describe a mobile visual search system that adopts the client-server architecture in which the database is stored on the phone. The system uses the BoW approach, and four different compact database methods are experimented and their performances are compared. Li et al. [22] propose an on-device mobile visual search system. The system uses the BoW approach with a small visual dictionary due to the memory limitation. Additionally, the most useful visual words are selected to decrease the retrieval time considering the processor limitation. Guan et al. [15] describe an on-device mobile image search system, which is based on bag-of-features (BoF) approach. The system uses

approximate nearest neighbor search to use high dimensional BoF descriptors on the mobile device with less memory usage. The search system also utilizes the GPS data from the mobile device to reduce the number of images to be compared. In our case, considering the potential application areas of our system (e.g., mobile product search), the database must be stored on the server side. For speeding up the query processing, feature extraction and query processing is run in parallel as the user is taking multiple photos of the query object (see Section 3.2).

Mobile devices have high user interaction potential; this has been utilized for better retrieval. Joint Search with Image Speech and Words (*JIGSAW*) [35] is a mobile visual search system that provides multimodal queries. This system allows the user to speak a sentence and performs text-based image search. The user selects one or more images from the result set to construct a visual query for content-based image search. In [30], a mobile product image search system that automatically extracts the object in the query image is proposed. From the top n images that have a clean background, object masks are found. The object in the query image is then extracted by using a weighted mask approach and its background is cleaned. The cleaned query image is finally used to perform image search. Extracting the object-of-interest and performing the query with a clean background is shown to work better. Similarly, *TapTell* [39] is an interactive mobile visual search system, in which users take a photo and indicate an object-of-interest within the photo using various drawing patterns. In our system, user interaction is used to obtain multi-view images of the query object. Further user interaction, e.g., to select the object of interest to suppress the effects of background, would further improve the performance as indicated by our experiments (see Section 5.2).

Landmark and location recognition and search are among popular mobile application areas [7,37,18,15,24,41]. In [24], 3D models of landmarks are constructed offline; then, low resolution query images taken by the mobile device are sent to the server and matched with 3D landmark models. With 3D models, it is possible to match query images taken from different viewpoints. However it is not easy to construct the 3D models of landmarks, especially large ones, because many views may be needed for a high quality reconstruction. In this work, we use the multi-view images directly, instead of building 3D models, since for a typical mobile product search system, multi-view images of products are readily available (but not as many views as would be needed to reconstruct a 3D model of the product). In [37,18], an *Active Query Sensing* system is proposed to help the user take a better query image when the first query fails, for mobile location search. The system learns the saliency of each view of a location to determine the discriminability, which is later used for suggesting better views based on the first query. Using multi-view queries, as in our system, might improve the accuracy at the first query and reduce the need to refine the search. However, the idea of selecting discriminative views is promising and can be investigated further for multi-view queries on multi-view databases to find out whether it is better than fusion approaches used in this work.

There are several mobile image search and recognition applications available on the mobile market. *Point&Find* [27] allows the users point the camera to the scene or object and get information about it. *Kooaba* is a domain-specific search system whose target domains are books, DVD and game covers [29]. *Digimarc Discover* [10] is similar to *Point&Find*; the user points the camera to an object and gets information about it. *PlinkArt* [9] is another domain-specific mobile search system whose target domain is well-known artworks. The user takes a photo of a well-known artwork and gets information about it. One of the latest mobile search application is *CamFind* [5], which is a general object search system. When the user takes a photo of a scene, products are identified and similar objects are listed as a result. Another recent mobile search application is *Amazon Flow* [1]; the user points the camera to the object and receives information about it. These examples indicate the commercial potential of mobile visual search systems.

Multi-image queries have been used to improve image retrieval. Arandjelovic and Zisserman [2] propose an object retrieval system using multiple image queries. The user enters a textual query and Google image search is performed using this textual query. The top eight images are then retrieved and used as query images. Early and late fusion methods are applied. Tang and Acton [33] propose a system that extracts different features from different query images. These extracted features are then combined and used as the features of the final query image. The system proposed in [25] allows users to select different regions of interest in the image. Then each region is treated as separate queries and their results are combined. Zhang et al. [38] describe a similar system, which also uses regions; however, these regions are extracted automatically and the user selects among them. Xue et al. [36] propose a system that uses multiple image queries to reduce the distracting features by using a hierarchical vocabulary tree. The system focuses on the parts that are common in all the query images. The multi-query system described in [21] uses early fusion; each database image is compared with each query image and each query image gets a weight according to the similarity between the query image and the database image.

All these works use multiple query images on single view databases for performance improvement; they do not utilize multi-view queries on multi-view databases. Moreover, a multi-view object image dataset to evaluate multi-view search systems is not publicly available. This paper aims to fill in these gaps.

3 Proposed Mobile Visual Search System

The proposed mobile multi-view visual search system is based on the well-known BoW approach: the images are represented as a histogram of quantized local features, called the BoW histogram. First, interest points are detected on images; the points are described with local descriptors computed around the points. Then, a vocabulary (dictionary) is constructed with a set of descriptors from the database, typically using the k-means clustering algorithm. Finally,

images are represented as a histogram of local interest point descriptors quantized on the vocabulary. When a query image is received by the search system, local features are extracted and BoW histogram is computed. The query histogram is compared with the histograms stored in the database, and the best k results are returned to the user (cf. Figure 2).

Local features are key to the performance of the search system. In a mobile system, they should also be efficiently computable. To this end, we employed two fast local feature detectors: Harris and Hessian [4,34]. They detect two types of complementary local structures on images: *corners* and *blobs*. Using complementary interest points are useful for improving the expressive power of features and hence the retrieval precision. The detected points are represented with the SIFT descriptor. The BoW histograms are computed for Harris and Hessian separately, and then they are concatenated to obtain the BoW histogram of an image.

For ranking, the database images need to be compared with the query image(s), based on the BoW histograms. It is crucial to select the right similarity functions for high retrieval precision and low computational cost. There are various similarity functions that can be used to compare histograms [19,20,23]. We experimented with the similarity functions given in Table 1 and presented a comparison in terms of retrieval precision and running time in Section 5. In the table, h^q and h^d represent the histogram of the query and database images, respectively. In the formulae, q_i and d_i are the i^{th} histogram bin of query and database histograms, respectively.

Table 1 Similarity functions for comparing BoW histograms.

Similarity Function	Symbol	Formula
Dot Product [20]	$dot(h^q, h^d)$	$\sum_i q_i d_i$
Histogram Intersection [23]	$HI(h^q, h^d)$	$\frac{\sum_i min(q_i, d_i)}{min(\|h^q\|, \|h^d\|)}$
Normalized Histogram Intersection [20]	$NHI(h^q, h^d)$	$\sum_i min\left(\frac{q_i}{\sum_i q_i}, \frac{d_i}{\sum_i d_i}\right)$
Normalized Correlation [23]	$NC(h^q, h^d)$	$\frac{\sum_i q_i d_i}{\sqrt{\sum_i q_i^2} \times \sqrt{\sum_i d_i^2}}$
Min-Max Ratio [6]	$MinMax(h^q, h^d)$	$\frac{\sum_i min(q_i, d_i)}{\sum_i max(q_i, d_i)}$

3.1 Multi-View Search

Image databases typically contain single view images of objects or scenes, as in Figure 3. At query time, if the user captures and uses a view close to the one in the database, she will retrieve the relevant image, but the user does not have any idea about the view stored in the database. If the query image has a slightly different view or scale, the invariancy of local features can handle such view/scale changes; but if the view/scale change is significant, the system will most probably fail. As a solution, the user may take multiple photos from different viewing angles and at different scales to increase the chance of providing query images similar to the database images. Moreover, if the database images are also multi-view, we can expect to get even better results. Hence, both the query and database images can be multi-view, each object/scene having multi-view images, as in Figure 4. In the most general case, the query may contain $M \geq 1$ images of an object and the database may consist of $N \geq 1$ images of each object.

Fig. 3 Single view images: each image is a typical, single view of an object.

Single-view query and single-view database ($M = 1$, $N = 1$): Both the query and database objects have a single image that represents a specific view of the object, as in Figure 3. During retrieval, the query image is compared to every database image using a similarity function (cf. Table 1) to find best k matches.

Single-view query and multi-view database ($M = 1$, $N \geq 1$): The query has single-view (cf. Figure 3) and database objects have multi-view images (cf. Figure 4). During retrieval, early/late fusion methods (cf. Sections 3.1.1 and 3.1.2) are employed to find and return best k matching database objects.

Multi-view query and single-view database ($M \geq 1$, $N = 1$): The query has multi-view images, the database has a single image for each object. During retrieval, early/late fusion methods are employed to find and return best k matching database images.

Multi-view query and multi-view database ($M \geq 1$, $N \geq 1$): Both the query and database objects have multi-view images. This is the most general case and comprises the previous three cases. During retrieval, early/late

Fig. 4 Multi-view images: each object has multiple images from different viewing angles (and/or at different scales).

fusion methods are employed to find and return best k matching database objects. We expect to get the best retrieval precision, but at an increased computational cost.

When the query or database objects have multiple images, we must employ fusion methods to process the queries and find best k matching database objects. This is one of the crucial steps to achieve high retrieval performance. There are mainly two types of fusion methods: *early fusion* and *late fusion*. We performed comprehensive experimental analysis of several early and late fusion methods.

3.1.1 Early Fusion

Early fusion, also referred to as fusion in feature space, is the approach in which the BoW histograms of multiple images are combined into a single histogram and the similarity function is applied on the combined histograms. We used the early fusion methods given in Table 2 [2]. In the table, the histograms for M images are combined into h^c; h_i^j is the i^{th} bin of histogram h^j of image j.

3.1.2 Late Fusion

Late fusion, also referred to as decision level fusion, considers each query and database image separately to obtain similarity scores between the query and database images using their BoW histograms; the final result list is obtained by combining the individual similarity scores. This can be done in two ways: (1) *image similarity and ranking* and (2) *image set similarity and ranking*.

Table 2 Early fusion methods.

Method	Formula
Sum Histogram	$h_i^c = \sum_{j=1}^{M} h_i^j$
Average Histogram	$h_i^c = \frac{\sum_{j=1}^{M} h_i^j}{M}$
Maximum Histogram	$h_i^c = max(h_i^1, \ldots, h_i^M)$

Image similarity and ranking. The image histograms in the query are compared to all the image histograms of all the objects in the database; a single result list is obtained by ranking the database objects based on similarity scores or ranking. We used the following methods [21,42].

- *Max Similarity (MAX SIM).* Each database image is compared with the query images and the similarity is taken as the maximum of the similarities.
- *Weighted Similarity.* Each database image is ranked according to a weighted similarity to the query images.
- *Count.* For multiple query images, multiple result lists are obtained. Then, for each image, a counter is incremented if it is in a list. Finally, the counter value is used to rank the database images (higher value, higher rank).
- *Highest Rank.* For multiple query images, multiple result lists are obtained and the highest rank is taken for each database image.
- *Rank Sum.* For multiple query images, multiple result lists are obtained and the ranking of each image in every list is summed and the resulting values are used to rank the database images.

Image set similarity and ranking. First, the similarity scores between M images of the query object and N database images of each object are computed, resulting in $M \times N$ similarity scores, as shown in Figure 5. Then, an *image set similarity score* between the query object and each database object is computed, and finally, database objects are ranked according to the image set similarity scores.

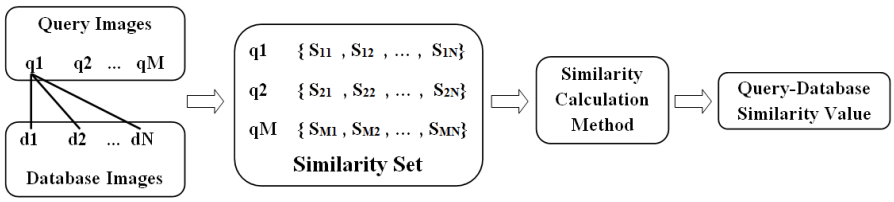

Fig. 5 Similarity computation between image sets. The query has M images, the database object has N images. A similarity score S_{ij} is computed between every query image i and every database object image j, resulting in $M \times N$ similarity scores.

The image set similarity scores between M query images and N database object images are computed in one of the following ways, based on the individual similarity scores between the query and database images (cf. Figure 5).

- *Maximum Similarity (MAX).* The similarity score is the maximum of all $M \times N$ similarity scores.

$$Similarity = max(S_{ij})$$

  If at least one of the query images is very similar to one of the database object images, this measure will return good results.
- *Average Similarity (AVERAGE).* The similarity score is computed as the average of all $M \times N$ similarity scores.

$$Similarity = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} S_{ij}}{M \times N}$$

  The average operator reduces the effects of outliers, but it also reduces the effects of good matches with high similarity scores.
- *Weighted Average Similarity (WEIGHTED AVERAGE).* To promote the influence of good matches with high similarity scores, a weight is assigned to each score.

$$W_{ij} = \frac{S_{ij}}{\sum_{i=1}^{M} \sum_{j=1}^{N} S_{ij}}$$

$$Similarity = \sum_{i}^{M} \sum_{j}^{N} S_{ij} \times W_{ij}$$

- *Average of Maximum Similarities (AVERAGE MAX).* First, the maximum similarity for each of M query images to N database object image is computed. Then, the average of M maximum similarity values is computed as the image set similarity.

$$Similarity = \frac{\sum_{i}^{M} \max(S_{i1}, \ldots, S_{iN})}{M}$$

- *Weighted Average of Maximum Similarities (WEIGHTED AVERAGE MAX).* This is similar to the previous method; this time, the average is weighted.

$$S_i = \max(S_{i1}, \ldots, S_{iN})$$

$$W_i = \frac{S_i}{\sum_{i}^{M} S_i}$$

$$Similarity = \sum_{i}^{M} W_i \times S_i$$

3.2 Speeding up Multi-View Query Processing

Multi-view queries are inherently computationally more expensive than single view queries. However, it is possible to speed up the multi-view search in a mobile multi-view search setting. As the user is taking multiple photos of the query object, the feature extraction and query processing can run in parallel in the background. This is possible because current mobile devices usually have multi-code processors. While one thread handles photo-taking, another thread can extract and send features to the server, which can start query processing as soon as it receives the features for the first query image. Figure 6 shows the flow diagram of the whole process as implemented in our mobile search system.

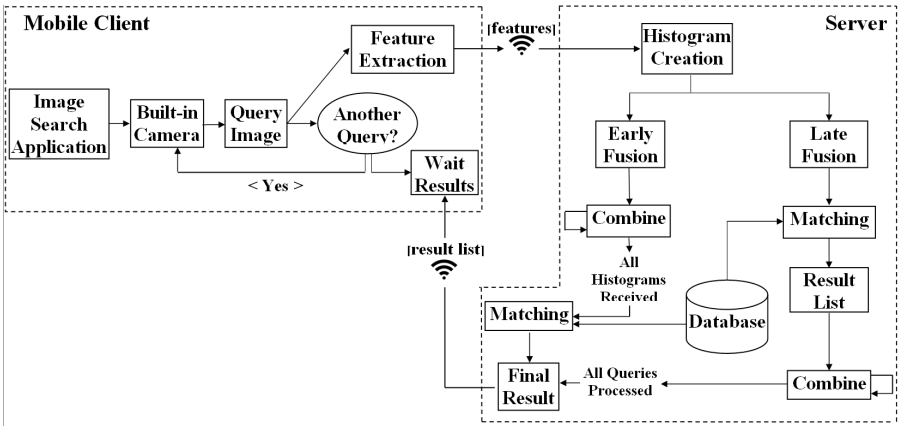

Fig. 6 Workflow of our image search system using early and late fusion methods.

4 Datasets

We used two different datasets to evaluate the performance of our mobile search system: (i) an existing single view mobile product image search dataset, *Caltech-256 Mobile Product Search Dataset* [30], and (ii) a new multi-view object image dataset we constructed for this work.

(i) Caltech-256 Mobile Product Search Dataset. This is a subset of the Caltech-256 Object Category Dataset [14], which is used to evaluate the performance of the mobile product search system described in [30]. The dataset has 20 categories and 844 object images with clean background; objects are positioned at the image center. There are 60 query images from six categories; query images contain background clutter. The original Caltech-256 dataset images were downloaded from Google Images. Figure 7 shows sample images from the dataset. This is a single view object image dataset. Although the dataset

contains multiple images of each category, the images are not multiple views of the same object, rather they are from different objects of the same category.

Fig. 7 Sample images from the *Caltech-256* mobile product search dataset.

(ii) Multi-View Object Images Dataset (MVOD). The main focus of this work is mobile multi-view object image search; hence it is crucial to have a suitable multi-view object image dataset to evaluate the performance of our system. To the best of our knowledge, such a dataset is not publicly available. We constructed a new dataset, called Multi-View Object Image Dataset (*MVOD 5K*), from online shopping sites. The dataset has 5000 images from 45 different product categories (shoes, backpacks, eyeglasses, cameras, printers, guitars, pianos, coffee machines, vacuum cleaners, irons, etc.). There are 1827 different object instances (from 45 categories) and each object has at least two different images taken from different views. On the average, there are 40 object instances and 111 images per category and 3 views per object. The images mostly have a clean background and objects are positioned at the image centers; this is on purpose, because the goal is to provide a good image of the product to attract the customers. The dataset is suitable for a mobile product search system, containing images of daily life items sold on online stores, and hence, it is easy to generate multi-view query images with a mobile device. Figure 8 shows sample images from the dataset. The dataset and more detailed description are available at `www.cs.bilkent.edu.tr/~bilmdg/mvod/`.

5 Experiments

We performed extensive experiments on the *Caltech-256* and *MVOD* datasets and evaluated the performance of various similarity functions and fusion methods. We used the OpenCV library [17] to extract the local features (Harris, Hessian detector with SIFT descriptor).

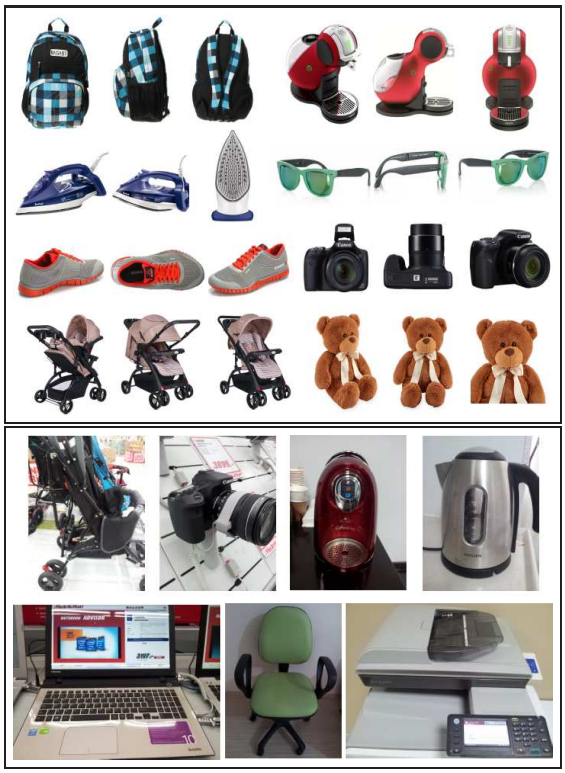

Fig. 8 Sample images from the *MVOD* dataset. Top: database, bottom: queries (selected single view images from multi-view queries).

The evaluation is done based on *average precision (AveP)* [16], as shown below. In the equation, k represents the rank in the list of retrieved images and N is the length of the list. A retrieved object image is relevant if it belongs to the same object category.

$$P(k) = \frac{relevant\ images \cap first\ k\ images}{k}$$

$$rel(k) = \begin{cases} 1, & \text{if } image\ k\ is\ relevant \\ 0, & \text{otherwise} \end{cases}$$

$$AveP = \frac{\sum_{k=1}^{N}(P(k) \times rel(k))}{N}$$

5.1 Results on the *Caltech-256* Dataset

As mentioned above, *Caltech-256* mobile product search dataset is a single-view dataset. On this dataset, we performed experiments with three types of queries:

- *Single view queries.* Each query is a single object image; the same query images, as in [30], are used (six categories, each having ten images). Queries with clean and cluttered background are performed and evaluated separately. We used clean background queries provided by [30]. They were obtained by segmenting out the objects from the background.
- *Multi-image queries.* Each query consists of multiple object images from the same category, however, the images belong to different objects, they are not multiple views of the same object. There are six queries for six categories, and all ten images are used in each multi-image query.
- *Multi-view queries.* Each query consists of multi-view images of an object; the images were taken with a mobile phone and hence not from the *Caltech-256* dataset. There are four multi-view queries for four categories, each having five images.

The vocabulary size is $3K$ and hard assignment is used for computing the BoW histograms. Figure 9 shows the average precision graphs for single view queries using various similarity functions. The similarity functions *Min-Max Ratio*, *Normalized Histogram Intersection* and *Normalized Correlation* work much better than *Dot Product* and *Histogram Intersection* on both clean and background cluttered queries. As expected, the average precision is higher for queries with a clean background. When we compare our results with those presented in Figure 6 (b) of [30], our average precision values are $0.1 - 0.15$ higher than [30], probably due to the multiple complementary features (Harris+Hessian with SIFT) we used. Figure 10 shows single view query examples with two different similarity functions.

Figure 11 shows the average precision graphs for multi-image queries using various fusion methods and the *Min-Max Ratio* similarity function. The late fusion methods *Rank Sum* and *Count* work better than the other early and late fusion methods. The average precision values are about 0.25 higher on background cluttered queries, and 0.1 higher on clean background queries, compared to the single view queries. Figures 12 and 13 show sample queries.

Figures 14 and 15 show the average precision graphs and a sample query for multi-view queries using various fusion methods and *Min-Max Ratio* similarity function. As explained above, the multi-view query images of objects were taken with a mobile phone on a clean background. The late fusion methods *Rank Sum*, *Weighted Similarity* and *Count* work better than the other early and late fusion methods. Multi-view queries improve the average precision performance further compared to multi-image queries, since the query images are multiple views of a single object, providing better representation for the query object.

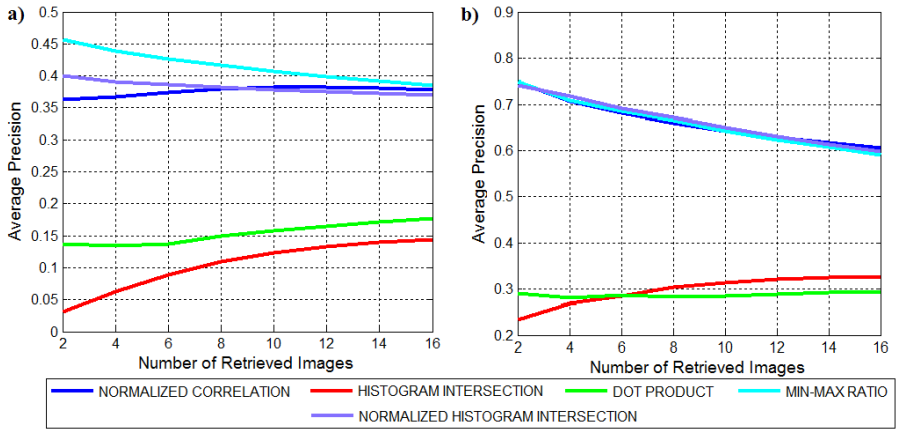

Fig. 9 Single view query average precision graphs on *Caltech-256* dataset with various similarity functions: (a) our results with background cluttered queries and (b) our results with clean background queries.

Fig. 10 Single view query examples on *Caltech-256* dataset with two similarity functions: (a) Min-Max Ratio and (b) Normalized Histogram Intersection.

5.2 Results on the *MVOD* Dataset

As mentioned above, *MVOD* is a multi-view dataset we prepared to evaluate the performance of our mobile search system on multi-view object image databases. It is much larger and more challenging than the *Caltech-256* dataset. Since this is a new and completely different dataset, the results are not directly comparable to those of *Caltech-256*. On this dataset, we performed single view and multi-view query experiments with two types of queries:

- *Internet queries.* The multi-view images of 45 queries, one query per category, are collected from online shopping sites. Similar to the *MVOD* dataset, these query images mostly have clean background.
- *Mobile phone queries.* The multi-view query images are obtained with a mobile phone in natural office, home or supermarket environments, in realistic conditions, having adverse effects, like background clutter and illumination problems. The query set has 15 queries for 15 categories.

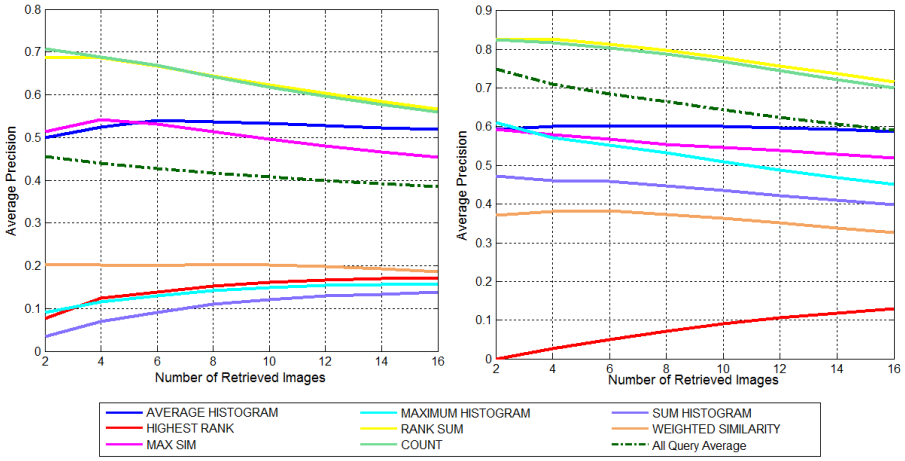

Fig. 11 Multi-image query average precision graphs on *Caltech-256* dataset with various early and late fusion methods and the Min-Max Ratio similarity function: (a) background cluttered queries, and (b) clean background queries.

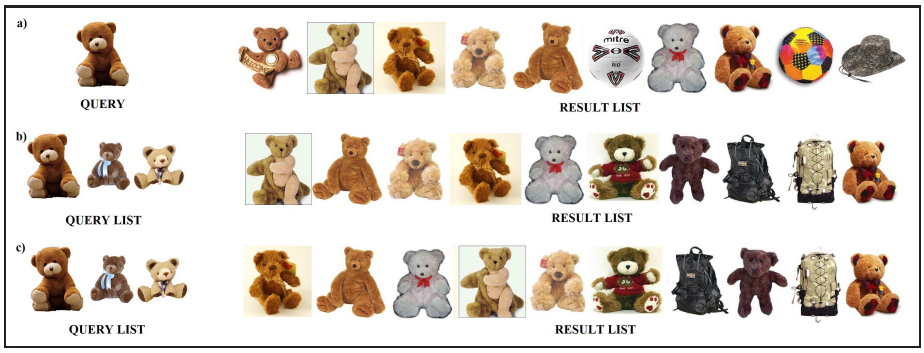

Fig. 12 Single view and multi-image query examples on *Caltech-256* dataset: (a) single view query, (b) multi-image query with *Rank Sum* late fusion method, and (c) multi-image query with *Count* late fusion method. The *Min-Max Ratio* similarity function is used.

Fig. 13 Multi-image query examples on *Caltech-256* dataset with early fusion: (a) Average Histogram, and (b) Weighted Average Histogram. The *Min-Max Ratio* similarity function is used.

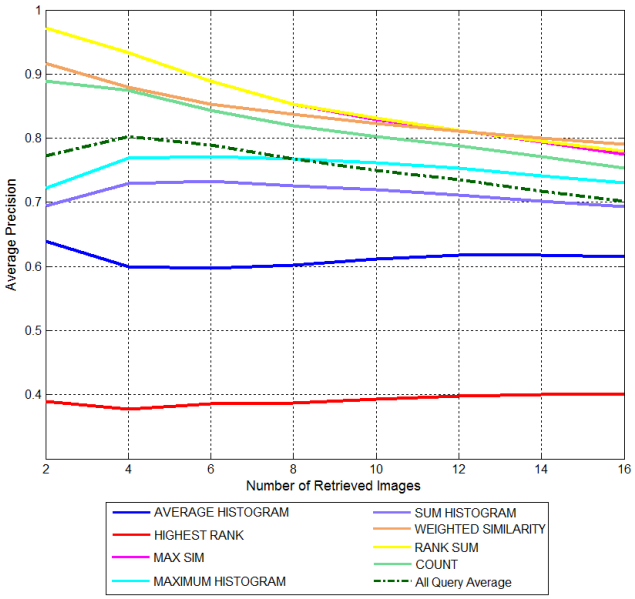

Fig. 14 Multi-view query average precision graph on *Caltech-256* dataset with various early and late fusion methods. The *Min-Max Ratio* similarity function is used. The multi-view query images were taken with a mobile phone.

Fig. 15 Multi-view query example on *Caltech-256* dataset with Rank Sum fusion and Min-Max Ratio similarity function. The query images were taken with a mobile phone.

The vocabulary size is $10K$ and hard assignment is used for computing the BoW histograms. Single view queries are performed by randomly selecting one of the query views and matching it with one of the database images. Multi-view queries are performed and presented for the best performing similarity function (*Min-Max Ratio*) and best early/late fusion methods based on the above experiments.

Figure 16 shows the average precision graphs for single view and multi-view queries on the *MVOD* dataset. As expected, the average precisions on *MVOD* is lower than those of *Caltech-256*. Parallel with the *Caltech-256* results, multi-view queries provide an improvement of about +0.1 to +0.2 over single view queries. The improvement is more on background cluttered queries (taken with a mobile phone), which is important, since, in a real world setting, the query images will usually have background clutter. On the other hand, the average precision for queries with clean background is always higher than queries with cluttered background. It is possible to reduce the influence of background by segmenting out the objects automatically, as in [30], or semi-automatically if

the user can quickly tap on the screen and select the object of interest, as in [39].

Among the fusion methods, the late fusion methods, *Weighted Average of Maximum Similarity* and *Maximum Similarity* work better than others. Sample queries in Figures 17, 18 and 19 demonstrate the improvement in the result lists for both clean background Internet queries and cluttered background mobile phone queries.

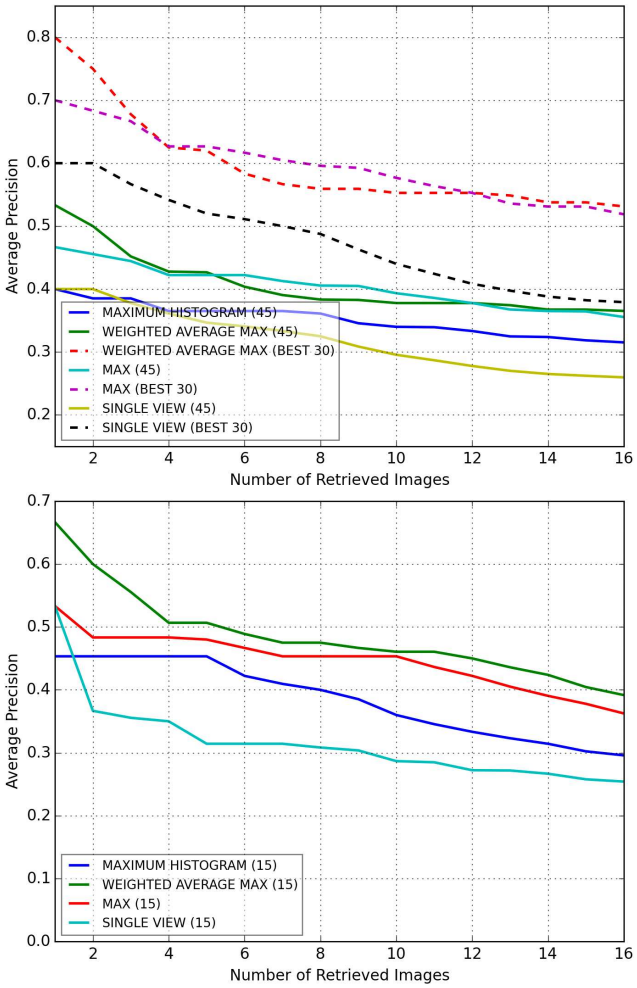

Fig. 16 Average precision graphs on the *MVOD* dataset with various early and late fusion methods. Top: query images are from the Internet. Bottom: Query images are taken with a mobile phone. The Min-Max Ratio similarity function is used. The numbers in the legends are the number of queries in the experiment.

Fig. 17 Single view and multi-view query examples on the *MVOD* dataset, multi-view query with *Max* late fusion method. The query images are taken with a mobile phone.

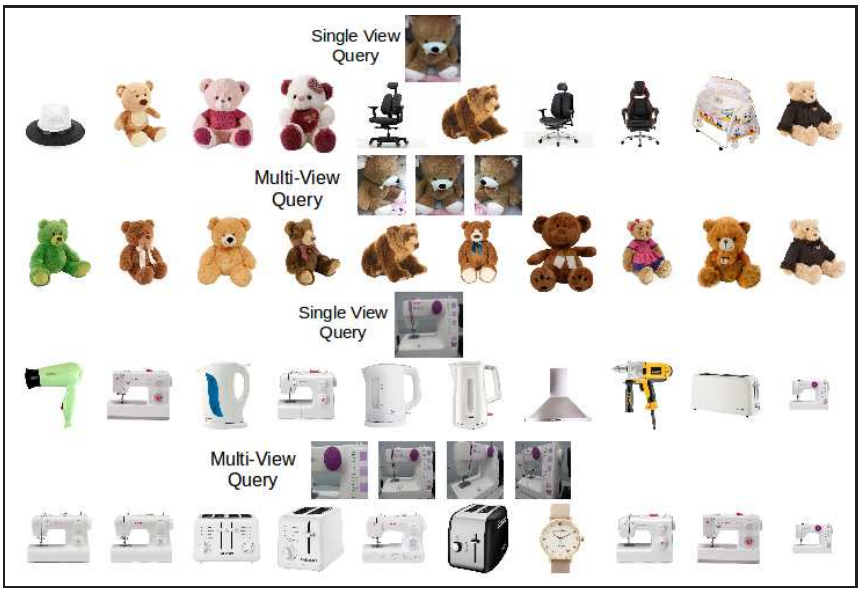

Fig. 18 Single view and multi-view query examples on the *MVOD* dataset, multi-view query with *Weighted Average Max* late fusion method. The query images are taken with a mobile phone.

Fig. 19 Single view and multi-view query examples on the *MVOD* dataset, multi-view query with *Max* late fusion method. The query images are from online shopping sites.

5.3 Running Time Analysis

Multi-view queries are inherently computationally expensive. In this section, we compare the running times of single and multi-view query methods for different similarity functions. To do so, we measured the time spent for matching the images on the server side; this includes the vector quantization, BoW histogram construction, similarity computation, fusion and ranking. The measurement is done on the *MVOD* dataset, with five queries each having five images, and the measured duration is the average of all queries in each (query) category. Table 3 summarizes the results. According to the table, the matching times for the similarity functions are close to each other. The increase in running time in multi-view queries is not proportional to the number of images in the query and database, it is lower (due to varying image content and different numbers of interest points detected in each image). Based on the running times and the average precision performances, the late fusion methods, *Weighted Average of Maximum Similarities* and *Maximum Similarity*, and the early fusion method, *Maximum Histogram*, can be used for multi-view object image search.

Table 3 Running times (ms) of similarity functions and fusion methods.

		Similarity Functions				
		Normalized Correlation	Histogram Intersection	Normalized Histogram Intersection	Dot Product	Min-Max Ratio
Fusion Methods	Single View Query (No Fusion)	226	208	242	258	212
	Sum Histogram	235	215	322	364	249
	Average Histogram	247	261	336	326	258
	Maximum Histogram	243	294	314	357	253
	Average Similarity	983	997	1211	958	994
	Weighted Avg. Sim.	996	1026	1150	973	1016
	Maximum Similarity	971	1015	1118	964	977
	Average of Max Sim.	976	1006	1125	961	981
	Weighted Avg. Max Sim.	967	1014	1147	976	986

6 Conclusions and Future Work

We proposed a new multi-view visual query model on multi-view object image databases for mobile visual search. We investigated the performance of single view, multi-image and multi-view object image queries on both single view and multi-view object image databases using various similarity functions and early/late fusion methods. We conclude that multiple view images, both in the queries and in the database, significantly improve the retrieval precision. As a result, mobile devices with built-in cameras can leverage the user interaction capability to enable multi-view queries. The performance can be further improved if the query objects are isolated from the background. This can be done automatically as in [30] or via user interaction, e.g., the user can tap on the screen and select the object-of-interest in the query image [39]. We implemented a mobile search system and evaluated it on two datasets, both suitable for mobile product search, which is one of the useful application areas of such mobile interactive search systems. Collecting and annotating a large-scale multi-view object image dataset remains as a future work.

Recently, deep convolutional neural networks (ConvNets) have proven to give state-of-the-art results in many computer vision problems, including image classification and retrieval [3,28]. Instead of keypoint-based BoW histograms, ConvNets features can also be used for retrieval in our multi-view object image search framework, since our framework is independent of the features used.

ConvNets features may be extracted on the mobile device and sent to the server, as in the current architecture, or multi-view images may be sent to the server and all the processing can take place on the server. High-performance ConvNets are usually quite large with millions of parameters [32] and require a high amount of processing power and memory. This is a serious limitation for a mobile search system; using large networks on current mobile devices is not feasible due to to the stringent memory limits on the running processes. Smaller networks, on the other hand, may not give satisfactory performance. The second alternative, sending images to the server, may require a large amount of uplink data traffic, which may be both costly and slow (upload data rates are much lower than download data rates). In summary, an interesting

research direction is to design a mobile search system architecture that can use the state-of-the-art ConvNets efficiently.

Acknowledgements The first author was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) under BİDEB 2228-A Graduate Scholarship.

References

1. A9.com, Inc.: Amazon Flow. `http://www.a9.com/whatwedo/mobile-technology/flow-powered-by-amazon` (2015). Accessed: 2016-03-02
2. Arandjelovic, R., Zisserman, A.: Multiple queries for large scale specific object retrieval. In: British Machine Vision Conference, pp. 92.1–92.11. BMVA Press (2012)
3. Babenko, A., Lempitsky, V.: Aggregating Deep Convolutional Features for Image Retrieval (2015)
4. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. In: European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 3951, pp. 404–417. Springer Berlin Heidelberg (2006)
5. CamFind: CamFind. `http://camfindapp.com` (2015). Accessed: 2016-03-02
6. Cha, S.H.: Comprehensive survey on distance/similarity measures between probability density functions. International Journal of Mathematical Models and Methods in Applied Sciences **1**(4), 300–307 (2007)
7. Chen, D.M., Baatz, G., Köser, K., Tsai, S.S., Vedantham, R., Pylvä, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., Girod, B., Grzeszczuk, R.: City-Scale Landmark Identification on Mobile Devices. In: Computer Vision and Pattern Recognition, pp. 737–744. IEEE (2011)
8. Chen, D.M., Girod, B.: Memory-Efficient Image Databases for Mobile Visual Search. In: IEEE Multimedia, vol. 21, pp. 14–23 (2013)
9. Cummins, M., Philbin, J.: PlinkArt. `http://www.androidtapp.com/plinkart` (2015). Accessed: 2016-03-02
10. DigiMarc, Co.: Digimarc Discover. `http://www.digimarc.com/discover` (2015). Accessed: 2016-03-02
11. Girod, B., Chandrasekhar, V., Chen, D.M., Cheung, N., Grzeszczuk, R., Reznik, Y.A., Takacs, G., Tsai, S.S., Vedantham, R.: Mobile Visual Search. IEEE Signal Processing Magazine **28**(4), 61–76 (2011)
12. Girod, B., Chandrasekhar, V., Grzeszczuk, R., Reznik, Y.A.: Mobile visual search: Architectures, technologies, and the emerging MPEG standard. IEEE MultiMedia **18**(3), 86–94 (2011)
13. Google, Inc.: Google Goggles. `http://www.google.com/mobile/goggles` (2015). Accessed: 2016-03-02
14. Griffin, G., Holub, A., Perona, P.: Caltech-256 Object Category Dataset. Tech. Rep. CNS-TR-2007-001, California Institute of Technology (2007)
15. Guan, T., He, Y., Duan, L., Yang, J., Gao, J., Yu, J.: Efficient BOF Generation and Compression for On-Device Mobile Visual Location Recognition. IEEE Multimedia **21**(2), 32–41 (2014)
16. Gunawardana, A., Shani, G.: A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. Journal of Machine Learning Research **10**, 2935–2962 (2009)
17. Itseez: OpenCV: Open Source Computer Vision Library. `http://opencv.org` (2015). Accessed: 2016-03-02
18. Ji, R., Yu, F.X., Zhang, T., Chang, S.F.: Active Query Sensing: Suggesting the Best Query View for Mobile Visual Search. ACM Transactions on Multimedia Computing, Communications, and Applications **8**(3s), 40 (2012)
19. Joseph, S., Balakrishnan, K.: Multi-Query Content Based Image Retrieval System using Local Binary Patterns. International Journal of Computer Applications **17**(7), 1–5 (2011)
20. Lampert, C.H.: Detecting objects in large image collections and videos by efficient subimage retrieval. In: International Conference on Computer Vision, pp. 987–994 (2009)

21. Lee, C.H., Lin, M.F.: A Multi-query Strategy for Content-based Image Retrieval. International Journal of Advanced Information Technologies **5**(2), 266–275 (2012)
22. Li, D., Chuah, M.C.: EMOD: An Efficient On-device Mobile Visual Search System. In: ACM Multimedia Systems Conference, pp. 25–36 (2015)
23. Mazloom, M., Habibian, A.H., Snoek, C.G.M.: Querying for video events by semantic signatures from few examples. In: ACM Multimedia Conference, pp. 609–612 (2013)
24. Min, W., Xu, C., Xu, M., Xiao, X., Bao, B.K.: Mobile Landmark Search with 3D Models. IEEE Transactions on Multimedia **16**(3), 623–636 (2014)
25. Moghaddam, B., Biermann, H., Margaritis, D.: Regions-of-interest and spatial layout for content-based image retrieval. Multimedia Tools and Applications **14**(2), 201–210 (2001)
26. Niaz, U., Merialdo, B.: Fusion methods for multimodal indexing of web data. In: International Workshop on Image and Audio Analysis for Multimedia Interactive Services. Paris, France (2013)
27. Nokia: Point and Find. `https://en.wikipedia.org/wiki/Nokia_Point_%26_Find` (2015). Accessed: 2016-03-02
28. Paulin, M., Douze, M., Harchaoui, Z., Mairal, J., Perronin, F., Schmid, C.: Local Convolutional Features with Unsupervised Training for Image Retrieval. In: International Conference on Computer Vision (2015)
29. Qualcomm Connected Experiences, Inc.: Kooaba: Image Recognition. `http://www.kooaba.com` (2015). Accessed: 2016-03-02
30. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Mobile Product Image Search by Automatic Query Object Extraction. In: European Conference on Computer Vision, pp. 114–127 (2012)
31. Su, Y., Chiu, T., Chen, Y., Yeh, C., Hsu, W.H.: Enabling low bitrate mobile visual recognition: a performance versus bandwidth evaluation. In: ACM Multimedia Conference, pp. 73–82 (2013)
32. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. arXiv preprint arXiv:1512.00567 (2015)
33. Tang, J., Acton, S.: An image retrieval algorithm using multiple query images. In: International Symposium on Signal Processing and Its Applications, vol. 1, pp. 193–196 (2003)
34. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: A survey. Foundations and Trends in Computer Graphics and Vision pp. 177–280 (2008)
35. Wang, Y., Mei, T., Wang, J., Li, H., Li, S.: JIGSAW: Interactive Mobile Visual Search with Multimodal Queries. In: ACM International Conference on Multimedia, MM '11, pp. 73–82 (2011)
36. Xue, Y., Qian, X., Zhang, B.: Mobile image retrieval using multi-photos as query. In: IEEE International Conference on Multimedia and Expo Workshops, pp. 1–4 (2013)
37. Yu, F.X., Ji, R., Chang, S.F.: Active query sensing for mobile location search. In: ACM International Conference on Multimedia, pp. 3–12. ACM (2011)
38. Zhang, C., Chen, X., Chen, W.B.: An Online Multiple Instance Learning System for Semantic Image Retrieval. In: IEEE International Symposium on Multimedia Workshops, pp. 83–84 (2007)
39. Zhang, N., Mei, T., Hua, X.S., Guan, L., Li, S.: TapTell: Interactive visual search for mobile task recommendation. Journal of Visual Communication and Image Representation **29**(0), 114–124 (2015)
40. Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N.: Query specific fusion for image retrieval. In: Computer Vision - ECCV 2012, *Lecture Notes in Computer Science*, vol. 7573, pp. 660–673 (2012)
41. Zhu, L., Shen, J., Jin, H., Xie, L., Zheng, R.: Landmark classification with Hierarchical Multi-Modal Exemplar Feature. IEEE Transactions on Multimedia **17**(7), 981–993 (2015)
42. Zhu, L., Zhang, A.: Supporting Multi-Example Image Queries in Image Databases. In: International Conference on Multimedia and Expo, pp. 697–700 (2000)