SURFLogo - Mobile Tagging with App Icons

Chadly Marouane[1](✉) and Andre Ebert[2]

[1] Virality GmbH - Research and Development,
Rauchstraße 7, 81679 Munich, Germany
marouane@virality.de
[2] Ludwig-Maximilians-Universität München,
Oettingenstraße 67, 80538 Munich, Germany
andre.ebert@ifi.lmu.de

Abstract. Mobile tagging became more and more popular in commercials, magazines, newspapers, and other applications during the last years. In context of commercials, a bar code containing the advertisers internet address is often used to refer a customer to related online content. Due to their robustness as well as their comparably high fault-tolerance in case of low quality pictures, QR-Code systems are commonly used for that task. Connected to that topic we present a special procedure for mobile tagging, which uses a distinct logo or image in order to refer to certain information instead of a QR-Code. Our procedure was optimized to work with a conventional smartphone – the only prerequisite for usage is the possession of a smartphone capable of capturing and analyzing the different logos with our smartphone application. To match the logos with related information and to determine their uniqueness we introduce a new similarity measure on basis of SURF feature points and a contour comparison.

Keywords: Mobile tagging · Mobile marketing · QR-code · Computer vision · App stores

1 Introduction

When talking of mobile tagging, an advanced process of associating real world items with digital information is meant [11]. In this context, this association represents an exciting as well as complex section in the area of current media developments [1,5,24]. Therefore, objects marked with tags are scanned with a smartphone's camera and the information encrypted within the tag can be processed. 2D codes, e.g., QR-Codes or Data-matrix codes are enriched with information which can be used and displayed on mobile devices like mobile phones, smartphones, or tablets. Hereby, new chances for enriching advertisements on posters, for pointing to localities on Google Maps, or to apps in app stores as well as for linking to profiles in social media networks are opened – the physical world becomes linked to its virtual counterpart. Contents of mobile tags are mostly conventional unique resource identifiers (URL) which get recognized

S. Sigg et al. (Eds.): MobiCASE 2015, LNICST 162, pp. 239–256, 2015.
DOI: 10.1007/978-3-319-29003-4_14

as the corresponding hyperlink though they are only represented by a simple string. Moreover, mobile tags also get used more and more for the advertisement of mobile apps by displaying a QR-Code in range of the app's print or its digital advertisement. As soon as a customer is curious about the app, he can become linked directly into the app store for downloading it.

But despite all of these advantages, there are major issues with the usage of QR-Codes. Besides their lack of an attractive visual appearance there is also a shortage of customer acceptance. A QR-Code often looks misplaced, especially when it takes up more space than the actual commercial. Related to that, there were developments for encoding additional information without a visual tag, e.g., under the usage of radio transmission technologies. Near Field Communication (NFC) is often used as a mobile tag in the tourism business [14], as well as in the smart home area [7] and the mobile payment sector [22]. Furthermore, the Bluetooth 4.0 standard became more attractive due to its energy efficient functioning. Technologies like Apple's iBeacons and Google's Eddystone are already deployed in the range of proximity marketing with great success [8,9].

But there are also disadvantages coming with these technologies. Compared to visual procedures, they tend to be expensive in purchase and their operational area is limited, e.g., they cannot be printed on a commercial poster or being displayed in a television advertisement. Another way for mobile tagging are Microsoft's Custom Tags. These are offering the possibility to use individual logos, brandings or photographies for mobile tags. Therefore, with the help of a specifically for this purpose developed treatment they are repainted with a matrix of dots. This treatment enables a special image scanner, which can be installed on a smartphone in form of an app, to recognize the custom tag [15]. Thus, the usage of QR-Codes can be prevented completely and the question is raised, if an app logo itself could be enough for linking additional content.

Driven by that idea and related to Microsoft's concept, we present an approach which enables us to identify apps distinctly by only analyzing their app logo. Similar to a bar code scanner for smartphones, a mobile application was developed to accomplish this. Therefore, we developed a multi-layered decisioning process for app logo identification, which is introduced and evaluated in the frame of this work. Additionally we present SURFLogoApp, a smartphone application consisting of a request server and a smartphone application which is capable of implementing our multi-layered decisioning process.

The contribution of this work is a system that can identify logos of an app store distinctly. Especially when using images of poor quality, taken by a smartphone camera, the system still returns a unique result. In this context, we present and evaluate a scanner that provides an easy and fast identification of app logos as well as a method that returns a unique result. Therefore, it uses a fine granular search and comparison process, a so-called multi-layered decisioning process. In contrast to a conventional search of images, such as Google Image Search, the method can also be applied to poor image quality. This paper is structured as follows: In Sect. 2 we explain some fundamentals of image processing relevant for the following application. Subsequently, we elucidate our general

concept in Sect. 3, consisting of the multi-layered decisioning process as well as the SURFLogoApp system. Section 4 evaluate our approach and provide some insights into our test results. Afterwards, we sum up our findings in Sect. 5 and give and outlook towards open issues and possible future work.

2 Image Processing

In the following we present some conceptual fundamentals concerning the main components of SURFLogo, i.e., image representation, feature point extraction and image comparison.

2.1 Representing Images with Feature Points

Many tasks and applications in the computer vision domain require a representation of images, which is detached from their raw pixels. For example, recognizing objects or identifying similar images based on raw pixel values gets unusable when images differ in color, illumination, scaling, or rotation. A solution to this problem is to compute so-called feature points which represent very characteristic and therefore highly distinctive points or areas of an image. Depending on the employed algorithm, feature points are robust against common transformations and varying contrast situations.

The Speeded Up Robust Features (SURF) algorithm by Bay et al. [3] can be used to compute feature points. It applies an approximation of the Gaussian blur filter to the image and then looks for local extrema to identify scale- and rotation-invariant feature points. Subsequently, these points are described via a 64-dimensional vector which, is computed from the Haar-Wavelet response of the feature point's surrounding region. Similarities between two feature points can be calculated with the help of the Euclidean Distance between them.

There are also techniques for identifying and describing feature points [6,17,18]. Recently, there is a trend towards binary descriptors (e.g., [2,4,13,19]), which can be compared to each other more efficiently (e.g., using Hamming Distance).

2.2 Image Comparison

In order to select the image out of the test set, which is most similar to a test candidate, image comparison techniques are necessary. All images are represented by feature points which are pre-computed in the first and become stored in a database in the following. In general, there are two different classes of matching techniques: Those working directly on feature points and those using so-called *visual words*.

Comparison Directly Based-on Feature Points. These approaches compare images using their respective feature points. Each feature point of the query image votes for a reference image out of the database, which contains the most similar feature point. The image with the most votes is selected as the one with the highest similarity [16,20].

A disadvantage of this concept is its inefficiency due to a huge amount of feature points to be considered as well as the high dimensionality of their descriptors. Even with moderately large databases, comparisons to a complete set of reference images can be unfeasible. Thus, a scalability to big amounts of data is not given.

Based on Visual Words. The second class of algorithms tries to overcome this shortcomings by virtually pre-computing matches of individual feature matching by quantizing feature descriptors. For that purpose, the existing feature points or a subset of them are clustered, e.g., with the k-Means algorithm. Thus, every feature point is assigned to its nearest cluster center. As a consequence, an image is represented by a histogram of cluster frequencies, whose comparison can be undertaken much more efficiently. Since most of these approaches originate from domains of text processing and document retrieval, the clusters are also called visual words, a set of all words is called the *vocabulary* and an image representation is called *bag-of-words*. One of the first approaches in that category was introduced by Sivic and Zisserman [21]. Additionally, Turcot and Lowe showed that it is possible to discard up to 96 % of all feature points without reducing the matching precision [23].

3 Concept

In the following we introduce our system's components as well as further information about these. Moreover, our multi-layered decisioning approach is explained in detail.

3.1 Control Concept

The core idea of SURFLogoApp is to link additional content, e.g., a hyperlink to an app store, to a physical as well as to a digital advertisement while completely resigning conventional methods like QR-Codes. Instead, we only use the apps own logo. The logo is detected and scanned by the smartphone's built-in camera and links the user to the commercials counterpart in the app store as well as it enables its download. In the following, we call this referencing logos SURFlogo.

The control concept of SURFLogo from the user's point of view is designed as follows (see Fig. 1): (1) the users recognizes a SURFLogo related to a specific app in a commercial, which is bounded by a quadratic twin framed box, (2) if SURFLogoApp was downloaded and installed on the user's device, the SURFLogo gets scanned with it and, (3) the user becomes redirected to an app store with the opportunity to download the associated app.

Fig. 1. Control concept of the SURFLogoApp - the SURFLogo which links to an associated app becomes scanned with the smartphone application

3.2 Components of SURFLogoApp

The distributed SURFLogoApp system consists of three different components, which are a request server, a database and a smartphone application. The communication between these components is established by the use of a conventional internet connection.

Database. The database components contains all registered SURFLogos in the form of a sequence of feature points $Seq_{Feature}$, a quantized vector of these feature points $Quant(Seq_{Feature})$ (*bag-of-words*), a sequence of contour points $Seq_{Contour}$, and a uniform resource identifier URL_{App} associated with an app in an app store.

$$AppIc_i = \begin{pmatrix} Seq_{Feature} \\ Quant(Seq_{Feature}) \\ Seq_{Contour} \\ URL_{App} \end{pmatrix}, i \in Database_{AppIc}$$

Furthermore, the database component contains a vocabulary, which enables the calculation of a quantized vector out of an image's sequence of feature points (bag-of-words). This procedure as well as the extraction of feature points and the contour extraction is undertaken for every SURFLogo in the database.

The vocabulary consists of all clustered feature points' cluster centers, which are available in the database. Related to that, the component can possess multiple vocabularies bound to specific areas of operation. The database's creation takes place in the so-called offline phase, scheduled prior to the system's operational phase. Still, the database can always be extended during the operational phase (online phase). On updating an existing SURFLogo, all related references in the database are also replaced.

Request Server. The request server receives and processes all requests for the SURFLogo system via a REST service.

A request consists of a sequence of feature points as well as of a sequence of contour points, which represent and describe a SURFLogo image. The sequences

are needed by the request server for searching the database with a matching algorithm. The algorithm's result is distinct and is represented by a direct hit or no existing associated representation. In case of a successful search, the server responses to the SURFLogoApp with an URL pointing to information related to the identified SURFLogo, else no result is returned.

Smartphone Application. The smartphone application's duty is to identify images marked as SURFLogos with its camera, to scan them and in the following to analyze them. In order to support a SURFLogo's distinct an automatic identification, it is marked with a quadratic, black, twin frame (see Fig. 2).

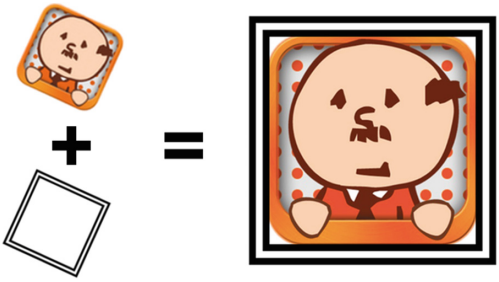

Fig. 2. Example of a SURFLogo – an app icon and a black twin frame for automated recognition by the smartphone's camera

For recognizing a SURFLogo, the frames needs to comply with the following requirements: (1) the corner points of one quadratic frame need to be within the other quadratic frame's corners, (2) the ratio of both of the quadratic frames needs to be within a defined area, and (3) both of the quadratic frames need to have a minimum size.

As soon as a SURFLogo becomes detected by a smartphone application, it gets scanned and rotated in dependence of its surrounding frames. The time needed to scan a SURFLogo is comparable to the time needed by a conventional QR-Code scanner.

After the frame became removed, all feature points are extracted out of the image and consolidated in a sequence $Seq_{Feature}$. In the next step, all two dimensional coordinates which are describing the images contour are also becoming aggregated in a sequence $Seq_{Contour}$. These sequences are now sent to the request server, which responds with an URL URL_{App} if the database search was successful. The URL references to the an apps download page within an app store.

3.3 Matching Algorithm

The matching algorithm serves for comparison of features and therefore distinct recognition of SURFLogos – it is one of the most important components

for the SURFLogoApp. During the comparison process it determines individual characteristics of a SURFLogo on basis of the given features. These characteristics allow a explicit distinction between different SURFLogos and false positive results in context of the searching process can be suspended. In a positive case, a distinct data entry related to the given features is found in the database, in a negative case no entry is available. The algorithm consists of two different steps: a preprocessing step followed by a matching step.

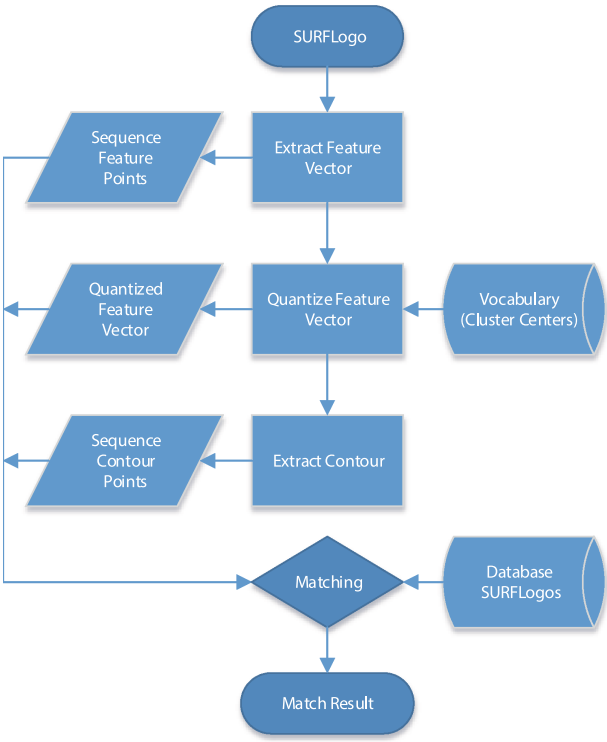

Fig. 3. Preprocessing: (1) Extraction of feature points and vectors, (2) quantization of feature vectors, (3) extraction of contour points. At least all information from (1), (2) and (3) will be used for the multi-layered decisioning process.

Preprocessing. The preprocessing step is needed during the offline phase at the database's creation as well as afterwards during the online phase. Especially for the extraction of an image's features and the SURFLogo's appending to the database, an effective preprocessing is crucial (see Fig. 3). Therefore, all feature points and their corresponding vectors are extracted out of a SURFLogo and saved temporarily. For the extraction routine the robust SURF process is used [3]. Subsequently, the extracted feature vectors become quantized with the help of a vocabulary which was created at the forefront. The quantization itself is

crucial for the process of searching in order to speed it up. The vocabulary is created by computing clusters, e.g., with the k-Means algorithm, out of a large range of feature points. Ideally this range represents the full number of all SURFLogos contained by the database. During the quantization each feature vector is matched with the cluster center it has the least distance to. Thus, each SURFLogo can be represented by a histogram of cluster frequencies. These histograms - so-called bag-of-words - are easy and efficient to compare to each other. Because of the fact that lots of these and related processing concepts are originated in text processing and document retrieval, the resulting clusters are also referred to as visual words, the number of all words is referred to as dictionary, and the associated image representation is referred to as bag-of-words. The quantized vector also gets saved temporarily. In the last step, the contour of the SURFLogo described by SURF feature points and consisting of x- and y-coordinates gets extracted and temporarily saved as a sequence. Afterwards, all temporarily saved data gets either transferred into the database for permanent storage together with an URL and the SURFLogo or is handed over to the multi-layered decisioning process for further analysis. During the online phase, requests from a smartphone application already contain feature and contour points, which is why the first and the third step can be skipped.

Matching: Multi-layered Decisioning Process. The multi-layered decisioning process is only necessary during the operational online phase. Thereby, temporarily saved information from the preprocessing step as well as the database are used for data input. All in all, the matching step consists of three comparing processes (see Fig. 4). The first comparing process considers the quantized vector. For this purpose, all quantized vectors out of the database are compared to the given quantized feature vector. The k closest possible matches are used for the second comparison process $k \in \{2, .., 25\}$. The value k is defined as a chosen threshold $\text{THRESHOLD}_{\text{Quant}}$, which is evaluated in Sect. 4.3. Otherwise, the k next possible matches are compared on basis of their feature points and are ranked descending by their number of successful matched features. For the last comparison process, which considers the contours of both of the entries, only the first two entries out of the ranked list are used. Is the number of feature matches for both entries below a chosen threshold $\text{THRESHOLD}_{\text{Surf}}$, the whole process is canceled again. Else, the sequences containing the contour points are analyzed by the Hausdorff distance. The winning sequence is the one with the smallest Hausdorff distance. Again, if the Hausdorff distance is above a chosen threshold $\text{THRESHOLD}_{\text{Hausdorff}}$ for both contours, the comparison process is canceled without a successful result.

4 Evaluation

In this section, we evaluate our SURFLogoApp system concerning its performance and provide detailed information about the evaluation's results.

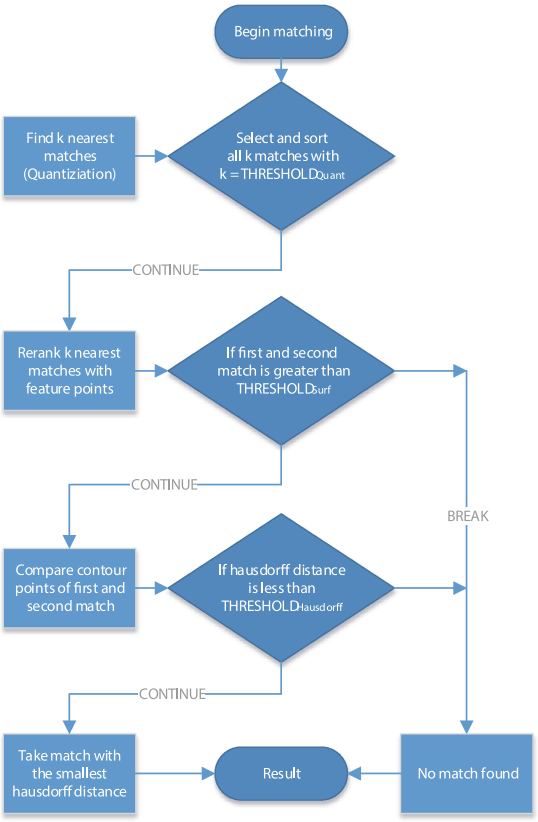

Fig. 4. Multi-layered decisioning process: (1) Find k nearest matches with quantized feature vectors, (2) rerank all matches with his feature points, (3) compare reranked matches with contour points.

4.1 Setup Configuration

We used one set of test data during the whole evaluation phase. Therefore, a database containing 5541 different SURFLogos was created. The logos itself were crawled and downloaded randomly out of Google's PlayStore and all in all we extracted more than 2118240 SURF feature points and stored them in our database.

A higher number of SURFLogos - so e.g., all 1.6 million logos from the Google PlayStore - has only a small influence on the duration of the query of the first sub-process of the multi-layered decisioning process and increases linearly with increasing number of SURFLogos.

In order to evaluate the multi-layered decisioning process, we generated 759 test images. In this context, a browser-based JavaScript application for sequential displaying of different SURFLogos in different sizes within one browser tab was developed. Additionally, the displayed SurfLogos were noted in a log file. The

counterpart of this setup is a modified version of the Android-based SURFLogoApp, which is capable of scanning the logos and storing them on a smartphone. Subsequently, the client-side Android application notifies the Javascript application via a REST-interface that a new picture was scanned and needs to be displayed in the browser. The test images were created by overall 6 volunteers equipped with a Samsung Galaxy S5 mini smartphone. The clustering calculations as well as the time measurement were conducted with a Mac Book Air 2013 (1,7 GHz Dual-Core i7, 4 MB on-chip L3 Cache, 8 GB 1600 MHz LPDDR3).

4.2 Critical Quantization Variables

The quantization of the feature vectors is dependent from different variables, which are important for a successful search process during the multi-layered decisioning process. That is why number and selection of feature vectors for generating a vocabulary as well as the number of words within a vocabulary are crucial for quantization. In general, quantization leads to a loss of information, which is why it is not sufficient to only consider the best suiting vector alone. Moreover, the next best candidates k also need to be examined. The correct result can be chosen out of this candidate range by a further selection step.

Subset Size and Feature Vector Selection. Because of the fact, that the vocabulary is crucial for the feature vector quantization, the following conditions must apply for the feature vector subset at the moment of the vocabulary creation: (1) the subset must be as big as possible; ideally it represents all feature vectors stored in the database and (2) it contains a broad spectrum of feature vectors; they are different compared to each other and their euclidean distance is as big as possible.

In order to create the vocabulary, the subset of feature vectors needs to be clustered. Therefore, a simple k-Means algorithm capable of clustering the whole subset is used [10]. When analyzing a large amount of vectors, placed in a vast vector space, e.g., a SURF feature vector with 64 dimensions, the clustering tends to be CPU- and time-intensive. That is why we evaluated different sizes of subsets while generating the vocabulary.

Table 1 shows the different time spans for subsets containing 10 %, 20 %, and 100 % of the existing feature vectors which complies to the whole amount of existing feature points.

Table 1. Time needed for clustering - 3 cluster sets with a subset of 10 %, 20 %, and 100 % from the database's feature vectors with a cluster size of 512.

Subset	10 %	20 %	100 %
Time	4 h 1 m 11 s	15 h 39 m 37 s	4d 12 h 2 m 29 s

The larger the vector amount for the vocabulary's generation is selected, the longer takes the process of clustering. The time span between the usage

of 10 %, which was 4 h, 1 min, and 100 %, which was 4 days, 12 h and 2 min, is extraordinarily significant. Furthermore, when examining the results from Fig. 5, it is clearly visible that the success rate of clusters generated with a subset of 20 % of all feature points is not particularly better than the one corresponding to clusters generated with a 100 %. In case of our 20 % subset, it is even worse. That is why for our evaluation we generated the clusters with a subset of only 10 % of all existing feature points from our database.

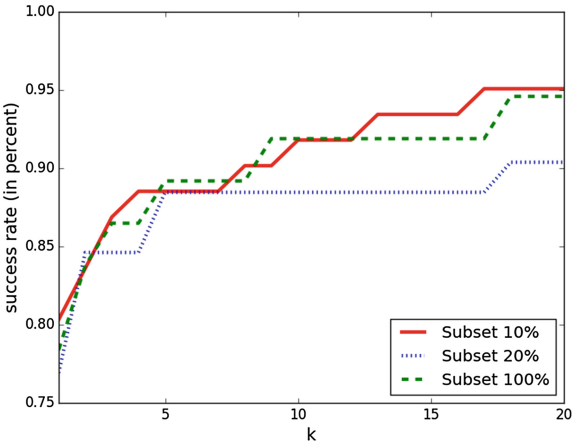

Fig. 5. Success rate of different vocabularies with increasing k – k is the number of possible result candidates. Vocabularies are created with a subset of 10 %, 20 %, and 100 % from the database's feature vectors and with a cluster size of 512.

Vocabulary Size. Besides the right amount of feature vectors for vocabulary generation, the number of clusters, which are representing a word, is also vital for the result's quality. If the size of a cluster is chosen to be to small, the loss of information due to the quantization process is larger and the quality of the results is smaller. In contrast, if the cluster size was chosen to be to large, the desirable time-saving due to the quantization process is narrowed.

Figure 6 shows, that with a rising number of clusters the result's quality is comparably high, even with a lower ranked k candidate. It is notable, that the discrepancy between the cluster sizes of 32 and 512 with $k = 1$ is already almost 30 %. Admittedly, it decreased with a growing k, but even with a candidate size of $k = 20$ it is still 10 %. Subsequently, we use a cluster size of 512 because of the superiority of its result's quality compared to cluster sizes of 32, 64, 128, and 256.

4.3 Multi-layered Decisioning Process

In context of our multi-layered decisioning process, we first examined some vital factors and variables within the procedure's sub-processes because of their

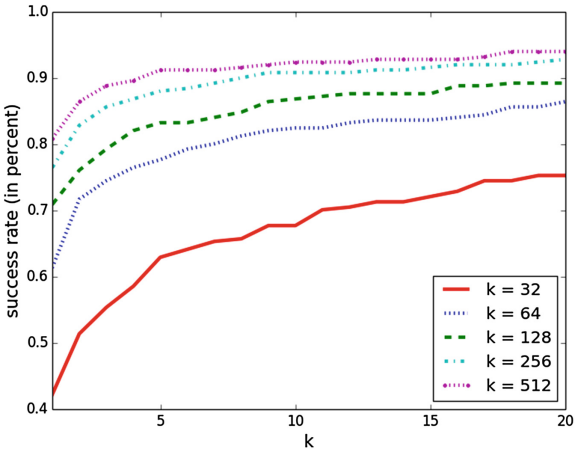

Fig. 6. Success rate of different vocabularies with increasing k - k is the number of possible result candidates. Vocabularies are created with a subset of 10 % from the database's feature vectors and with a cluster size of 32, 64, 128, 256 and 512.

influence onto our final results. Later on, we present an evaluation covering and examining our whole system and the quality of its results.

Quantization: Numbers of *k* Candidates. The identification of the right number of k candidates, which are the output of the quantization process, influences the following sub-process of feature vector comparison significantly. Moreover, the number of k candidates is heavily influencing the search and comparison time while matching the features of the SURF vectors. Because of the fact that all feature vectors are compared for each single candidate, a high number of candidates can slow down this sub-process noticeable. That is why it is important to identify the right amount of candidates in order to acquire an accurate result within an appropriate time span.

Figure 6 shows distinctly, that an increasing k also raises the success rate, independently from the chosen cluster sizes. The success rate itself already converges towards a maximum value for all different cluster sizes if $k = 18$. E.g., for a cluster size of 512, an amount of $k = 18$ candidates converges to an average of 94 %. That is why we use a candidate number of $k = 18$ with an additional buffer of 2 for the following evaluation, so we use all in all a $k = 20$. This k represent the $\text{THRESHOLD}_{\text{Quant}}$ like in Sect. 3.3 is described.

SURF Feature Matching. The following sub-process is a comparison search based on the actual SURF feature vectors. Therefore, the SURF feature vectors of all k candidates, which where determined in previous subprocesses are examined. In the easiest case all SURF feature vectors of the SURFLogo we want to analyze are compared to all k candidates and their feature vectors, respectively.

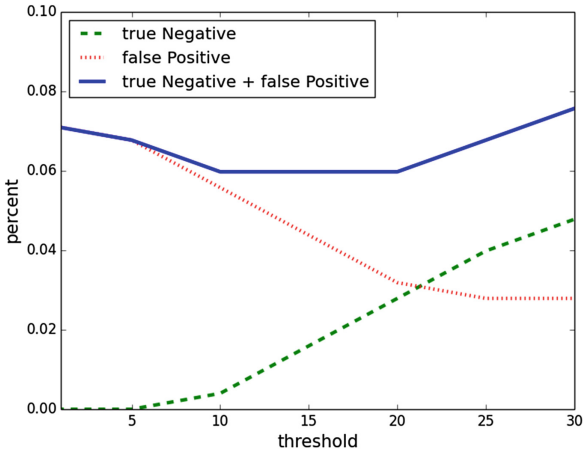

Fig. 7. Threshold's evaluation with different values of 0, 5, 10, 15, 20, 25 and 30. The sum of false positives and true negatives has its minimum between a threshold of 10 and 20. In order to have a small number of true negatives, a threshold of 10 is most ideal.

In contrast to the comparison process used during the quantization, the procedure used at this step of the analysis is much more complex. In that context, the complexity of the quantization's comparison procedure is about $O(1)$, the complexity of SURF's comparison procedure is $O(n)$. For that reason, a small number of comparison candidates is desirable at this step. As stated before, we use a candidate input of $k = 20$ for our evaluation. After comparing the 20 candidates on basis of their SURF feature descriptors, they are sorted anew with the goal of bringing the candidate with the most similar feature descriptors to the top. Our results are, that the correct logo is placed on the first two positions with a probability of 93 %, the correct result is on top of the list with a probability of 91 %. All in all it turned out, that after a quantization and the subsequent comparison search under the usage of feature points leads to a positioning of correct results in first place with a probability of 99 %. That is why the input for the last sub-process is limited to the first two results of the newly ordered candidate list. This limitation has positive effects in regarding the last sub-process, which is the process with the most computational costs. Furthermore, the 1 % of results ordered aback in our list can be neglected because of their little impact onto the overall result.

In order to better the result additionally, all possible false positives are about to be eliminated in the forefront. Therefore, candidates become rated false positive as soon as the sum of the number of all matched feature points is below a threshold of 10. This value represent the $\text{THRESHOLD}_{\text{Surf}}$ like in Sect. 3.3 is described.

An additional evaluation showed, that a value of 10 is already sufficient in order to exclude as much false positive candidates as possible. Figure 7 provides

information about a suitable threshold's evaluation with different values of 0, 5, 10, 15, 20, 25 and 30. It becomes apparent, that a threshold of 10 is suitable to sort out only a few number of valid candidates, but a large number of false positives and at the same time a small number of true negatives. If there are no candidates below this threshold, the procedure is canceled without a return value, which also means that the whole multi-layered decisioning process terminates.

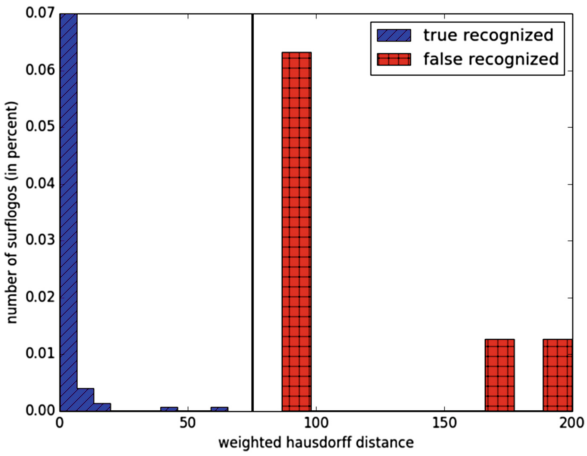

Fig. 8. Histogramms of all computed hausdorff distances: all distances with a true recognition are placed below a threshold of 75 (Black vertical line). Except for of a few outliers, the most of the true recognized results are in a range below a distance of 5.

Contour Detection and Hausdorff Distance. The last and thereby third sub-process consists of a contour comparison between two SURFLogos. The contour comparison itself is realized by using the Hausdorff distance [12]. In that context it measures the distance between two contours in a metric space, which allows us to compare them. Goal of this sub-process is the identification of the candidate among all candidates with the least contour distance – it is winning the comparison process and regarded as the correct result. For the contour we use the x- and y- coordinates of the SURF feature points which where matched in both images. With this approach it is guaranteed, that contour points created by noise or other camera effects are not included in the computation of the Hausdorff distance.

In order to prevent a better outcome for pairs with a small amount of common feature points, the distances itself become normalized. Thereby the Hausdorff distance is calculated as follows.

$$\frac{\text{distance}_{\text{hausdorff}}}{\#\text{matches}_{\text{featurepoints}}} \tag{1}$$

If there is no common feature point for a pair of candidates which is about to be compared, a maximum distance $\text{MAX}_{\text{float}}$ is returned.

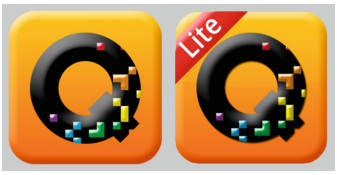

Fig. 9. A free and a purchasable version of an app logo. The hausdorff distances are really similar and difficult to distinguish.

Fig. 10. Example of a printed SURFLogo on the left side and a scanned SURFLogo with poor quality on the right side.

An evaluation on basis of this defined comparison procedure provided the following results. Figure 8 displays distinctly, that all distances are placed below a threshold of 75. So we set this value as $\text{THRESHOLD}_{\text{Hausdorff}}$ like in Sect. 3.3 is described.

Moreover, SURFLogos with small distance differences between candidate one and two are only logos which differ in small details. Figure 9 shows an example for those, where an app logo exists two times for a free and also for a purchasable version of the same application. This means, that in general the same app is referenced, but there are two existing versions of it. One which is about to be payed for and one which is available for free.

On basis of this insights, the third sub-process was extended by the following conditions:

- All distances placed above a threshold of 75 are neglected and not considered as a possible result.
- If the difference of the two comparison pair's distances is within a range from 0.5 to 1, both candidates are considered to be the correct SURFLogo.

Hence, the whole decisioning procedure of the sub-process is defined as follows: (1) the distances of the candidates 1 and 2 to the SURFlogo are computed. If both distances are placed above of a defined threshold in (2), the whole process terminates and no result is returned, else it continues. If the distances of two comparison pairs are placed within a range from 0 to 1 in (3), the process terminates and both associated candidates are returned, else the candidate with the smallest distance to the logo on the captured image is returned.

Results. After we reviewed the three sub-processes of our multi-layered decisioning process individually, the evaluation of the complete procedure is provided in the following. Therefore, the identified thresholds of the individual processes were used, the candidate pool delivered from sub-process 1 to sub-process 2 and from sub-process 2 to sub-process 3 was kept consistent.

All analysis requests with the multi-layered decisioning process are resulting a correct response in 92 %.

If the results for apps which possess two logo versions, e.g., for a free version and a purchasable version, are also taken into account the overall result rises to 93 %. Another significance of the overall result is the fact that there are no false positives occurring in the result quantity. This means, in the best case a distinct result is delivered by the multi-layered decisioning process, in the worst case no result is returned.

5 Conclusion

In the context of this paper we presented a procedure capable of replacing QR-Codes in their functionality of mobile tagging for commercials in print and digital media. Instead, the applications logo itself is used for automated identification and referral to the app store. For realization we introduced and evaluated a multi-layered decisioning process. Furthermore, we provided a distributed system called SURFLogo, which implements the procedure in form of a mobile application, a request server and a database, which supports the process's special requirements.

The evaluation of all three subprocesses of our multi-layered approach was undertaken separately for each subprocess. The first subprocess indicates, that a cluster size of 512 is suitable for the feature vectors' quantization and in order to reach a success rate of 82 % when searching for the associated SURFLogos. Because of the fact that such a result is not sufficient, we use the 20 best result candidates from subprocess 1 as input for subprocess 2. The second subprocess shows, that a search based on basis of feature points results in a better result set, which enables us to only use the two best candidates out of the second step for the third subprocess. Additionally, we introduced a threshold, which considers the sum out of successful created feature points for both candidates and cancels the process without return value if this value is placed below 10. The third subprocess identifies the correct candidate on basis of the SURFLogo contour. Therefore, we calculated the Hausdorff distance on basis of the x- and y-coordinates of all matched feature points and returned the candidates with the least distance as the correct result. In that context, we used another threshold capable of identifying distances larger than 75 as belonging to wrong candidates.

All in all, the multi-layered decisioning process has a success rate of 93 %. In 7 % of all cases we were not able to identify the right SURFLogo. There was no case of a false positive result being returned, which proves that the procedure is relatively distinct. Compared to the QR-Code procedure with a success rate of nearly 100 %, our approach is still very robust. Additionally it is based on a visually more individual content, which is easy assignable to a specific app. In contrast to image searching approaches (e.g., Google Image Search), a SURFLogo with a comparably poor quality is still identifiable (Fig. 10).

Despite the high success rate, there were some SURFLogos, which were not detected. Reasons for that may be a bad image quality due to the smartphone's camera, which can lead to interferences and inaccuracies due to noise or to blur effects conditioned by ambiguous movements. Furthermore, the procedure is colorblind, which means that it is not capable of distinguishing between SURFLogos

with identical content displayed in different contrasts or intensities of colors. An advantage of that is a higher robustness during different lighting conditions. However, because of that there are restrictions regarding the discriminability of similar contours. Thereto, the procedure could be extended by a color detector.

References

1. Al-Khalifa, H.S.: Utilizing QR code and mobile phones for blinds and visually impaired people. In: Miesenberger, K., Klaus, J., Zagler, W.L., Karshmer, A.I. (eds.) ICCHP 2008. LNCS, vol. 5105, pp. 1065–1069. Springer, Heidelberg (2008)
2. Alahi, A., Ortiz, R., Vandergheynst, P.: Freak: fast retina keypoint. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2012), pp. 510–517 (2012)
3. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
4. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: BRIEF: binary robust independent elementary features. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 778–792. Springer, Heidelberg (2010)
5. Canadi, M., Hpken, W., Fuchs, M.: Application of QR codes in online travel distribution. In: Gretzel, U., Law, R., Fuchs, M. (eds.) Information and Communication Technologies in Tourism 2010, pp. 137–148. Springer, Vienna (2010)
6. Chandrasekhar, V., Chen, D.M., Lin, A., Takacs, G., Tsai, S.S., Cheung, N.-M., Reznik, Y., Grzeszczuk, R., Girod, B.: Comparison of local feature descriptors for mobile visual search. In: 17th IEEE International Conference on Image Processing (ICIP 2010), pp. 3885–3888 (2010)
7. Darianian, M., Michael, M.: Smart home mobile RFID-based internet-of-things systems and services. In: International Conference on Advanced Computer Theory and Engineering, ICACTE 2008, pp. 116–120, December 2008
8. Gast, M.S.: Building Applications with iBeacon: Proximity and Location Services with Bluetooth Low Energy. O'Reilly Media, Sebastopol (2014)
9. Goosen, C.A.: Design and implementation of a bluetooth 4.0 le infrastructure for mobile devices, June 2014
10. Hartigan, J.A., Wong, M.A.: A K-means clustering algorithm. Appl. Stat. **28**, 100–108 (1979)
11. Hegen, M.: Mobile Tagging: Potenziale für das Mobile Business. Diplom.de (2010)
12. Huttenlocher, D., Klanderman, G., Rucklidge, W.: Comparing images using the hausdorff distance. IEEE Trans. Pattern Anal. Mach. Intell. **15**(9), 850–863 (1993)
13. Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: binary robust invariant scalable keypoints. In: IEEE International Conference on Computer Vision (ICCV 2011), pp. 2548–2555. IEEE (2011)
14. Madlmayr, G., Scharinger, J.: Neue dimension von mobilen tourismusanwendungen durch near field communication-technologie. In: Egger, R., Jooss, M. (eds.) mTourism, pp. 75–88. Gabler (2010)
15. Microsoft: Mircosoft Tag - Creating Custom Tags (2011). http://tag.microsoft.com/what-is-tag/custom-tags.aspx. Accessed 16 July 2015
16. Mikolajczyk, K., Schmid, C.: Indexing based on scale invariant interest points. In: Proceedings of Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 1, pp. 525–531. IEEE (2001)

17. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2005)
18. Miksik, O., Mikolajczyk, K.: Evaluation of local detectors and descriptors for fast feature matching. In: 21st International Conference on Pattern Recognition (ICPR 2012), pp. 2681–2684 (2012)
19. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: IEEE International Conference on Computer Vision (ICCV 2011), pp. 2564–2571 (2011)
20. Schaffalitzky, F., Zisserman, A.: Automated scene matching in movies. In: Lew, M., Sebe, N., Eakins, J.P. (eds.) CIVR 2002. LNCS, vol. 2383, pp. 186–197. Springer, Heidelberg (2002)
21. Sivic, J., Zisserman, A.: Video google: a text retrieval approach to object matching in videos. In: Proceedings of Ninth IEEE International Conference on Computer Vision, pp. 1470–1477 (2003)
22. Tan, G.W.-H., Ooi, K.-B., Chong, S.-C., Hew, T.-S.: NFC mobile credit card: the next frontier of mobile payment? Telematics Inform. **31**(2), 292–307 (2014)
23. Turcot, P., Lowe, D.G.: Better matching with fewer features: the selection of useful features in large database recognition problems. In: IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops 2009), pp. 2109–2116 (2009)
24. Walsh, A.: Blurring the boundaries between our physical and electronic libraries. Electron. Libr. **29**(4), 429–437 (2011)