AUTOMATIC RECOGNITION OF MULTIPLE BRANDS IN IMAGES ON MOBILE DEVICES

Marcin Skoczylas

Faculty of Computer Science, Bialystok University of Technology, Białystok, Poland

Abstract: Visibility of the product on a shelf is an important task of modern marketing principles. Very often companies have agreements with merchants that particular product will be visible and cover defined percentage of the shelf. This trivial task of counting the amount of products or branding logos that are visible within a certain range, is performed manually by an auditor that checks if the agreement is fulfilled. Up till now there does not exist an easy, mobile mechanism that allows to easily capture, recognise and count defined, multiple objects that are visible in the surroundings of the user. Such scenario however, can be achieved using modern mobile phones and their cameras to capture surroundings, and then use their computing power to perform the recognition and counting. For this purpose, feature detectors (such as SIFT, SURF or BRISK) are utilised to create a database of products box images and extracted keypoints are stored. In a new image keypoints are found using the same feature detector, but to avoid problem of multiple identical keypoints, the image is divided and analysed using a sliding window. Keypoints from a window are extracted and are considered as candidates for keypoints that correspond to training images. When enough points are found then perspective transform is calculated. If detected corners are correctly shaped then product is marked with recognised class. In this paper preliminary results of a mobile framework that allows recognition and counting of visible products in surroundings of the user will be presented.

Keywords: visual shelf monitoring product cover logo recognition keypoints detection mobile devices

1. Introduction

Visual shelf monitoring is a known method of checking products availability. There exist commercial applications that are able to monitor goods on the shelf and raise alarms when certain product is not available[1]. Such applications are relying on a simple detection mechanisms, such as comparing two adjacent images captured in two

Advances in Computer Science Research, vol. 10, pp. 81-97, 2013.

[1] for example http://mathecsys.com/visual-shelf-monitoring/

different moments of time, and raise alarms when an empty space is visible instead of a package or branding logo. This technology however, requires a static camera attached that is monitoring space only to a limited extent. Any angle changes or camera movements require image fetching calibration.

In a modern world there exists a requirement to perform image analysis in a fast and mobile way. As an example let's consider a typical check of the visibility of the product availability in shops, especially counting of products visibility percentage.

Visibility of the product on a shelf is an important task of modern marketing principles. Very often companies have agreements with merchants that particular product will be visible and cover defined percentage of the shelf. To confirm that the agreement is fulfilled the auditor is checking several shops and shelves every day. To check that the product is visible within a set range, this trivial task of counting the amount of products visible is performed manually by a person that checks if the agreement is fulfilled. Recent mobile phones are gaining computing power and can perform such tasks automatically.

Until now there does not exist an easy, mobile mechanism that allows a user to easily capture, recognise and count defined objects that are visible in the surroundings. Such scenario however, can be achieved using modern mobile phones and their cameras to capture surroundings, and then use computing power to perform the recognition and counting. In this paper preliminary results of a framework that allows recognition and counting of visible products in surroundings of the user will be presented.

2. Requirements

In this section main purposes of the presented approach, as well as differences in existing applications will be delineated.

The main purpose of the solution is a possibility of proper recognition and counting products or visible branding logos in user surroundings. Shapes, representing branding logos, are defined before capturing images on-site and are reused in following runs of the algorithm. The idea of this approach is that the auditor will use only a recent mobile phone (such as Android or iOS powered devices) with the recognition software installed. The auditor should be able to capture surroundings using his mobile phone camera, and as a result of the procedure, the algorithm should give to the user information about detected branding logos and visibility percentage. These values can be stored for further processing by the auditor or merchant. In overall the algorithm has to be robust with low computational complexity, but also with high accuracy. It's important to note, that packages and branding can (and certainly will)

be visible in user's surroundings in different scales and rotations. The solution should handle this situation gracefully, giving to the user possibility of different capture angles and light intensities, possibly without complicated calibration.

There exist different approaches to image recognition problems, using texture features classifiers [1], image processing filters, etc. Existing solutions can perform correct image detection and recognition in captured images as static [2], but also moving images [3]. None of existing commercial solutions of visual shelf monitoring or products visibility counting can capture and perform brand logo recognition in multiple images of user surroundings. Also the algorithms are aimed towards detection and recognition of only one object visible in the image. The problem described in this paper requires detection of multiple objects visible on multiple images.

3. Related work

Recognition of defined images on other images is a well-known problem and many solutions are existing. Most popular approach is to extract from training images and the captured image keypoints that are defining certain properties of the area. Different keypoint extracting algorithms were invented by authors over last many years, these include for example Scale-Invariant Feature Transform (SIFT [4]) or Speeded Up Robust Features (SURF [5]), but also recently presented Binary Robust Invariant Scalable Keypoints (BRISK [6]) and many others.

Topic of images recognition using keypoints descriptors became recently very popular. For example, SURF algorithm was successfuly utilised for face recognition [7] and road obstacle recognition [8]. The BRISK algorithm was implemened into traffic sign recognition as presented in [9]. These objects are recognised using a keypoints detector and descriptors of the points from training images are used to find similar descriptors in a captured image. One of such examples is also implementation of SIFT object recogniser in FPGA presented recently [10]. Authors implemented their algorithm using some of the concepts that are also a base for this publication, however the difference is that the algorithm from [10] can correctly identify only one object in the image. It's not suitable for counting amount of the same objects visible on captured image, but the possibility of real-time calculations thanks to robust FPGA implementation are promising.

There exist also different approaches to logo recognition as for example recently presented in [11]. Authors are classifying vehicle logos using SVM based on a Bag-of-Words (BoW) approach. The algorithm extracts SIFT features and quantizes features into visual words by 'Soft-assignment'. The database is then used for new objects recognition.

All the algorithms above are mainly searching for one particular object in the image. The purpose of this publication is definition of an algorithm, that can recognise multiple objects in the image, also repeating ones.

4. Recognition algorithm idea

4.1 Feature descriptors

Image feature descriptors are becoming a standard in current state of the art of image recognition algorithms. Their application is mainly for detection and recognition purposes, however there are additional tasks such as medical image registration [12]. For this study, author selected most common and popular feature detectors: SIFT, SURF, but also recently presented BRISK. Main principles of these three algorithms are delineated in the following paragraph.

- Scale-Invariant Feature Transform (SIFT)

The SIFT algorithm is quite computational extensive algorithm that detects features in images. Best thing about SIFT is that it is invariant to scale changes, but also rotations and light intensity changes. The main algorithm can be boiled down to these steps:

1. Scale space is constructed. The original image is incrementally convolved with gaussian operator: $L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$ multiple times, with different σ value[2] and then the resulting images are scaled down by a half. On scaled images, again the gaussian convolution is applied and so on. This operation generates multiple scale pyramids, with different σ parameters.
2. Local extrema are detected. Algorithm iterates for each pixel, that is compared to a current image, one above and one below in the pyramid scale. A key point candidate is detected only if it is larger than all neighbors or smaller than all others.
3. Key point candidates are evaluated. Low-contrast features and edges (such as corners) are removed.
4. Orientation is assigned to each key point. Gaussian smoothed image is selected that has the same scale as the key point. For each image pixel $L(x,y)$ at this scale, the gradient magnitude, $m(x,y) = \sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$ and orientation

[2] suggested amount of octaves by SIFT author is 5

$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$ are calculated around the key point. Orientation histogram is created with 36 bins (divided orientations in 360°). Then, from the histogram a main gradient orientation is identified by finding two preminent peaks. This key point is marked with that found gradient orientation (thus orientation invariance can be applied by removing that relative orientation).

5. Descriptor of the key point is generated. Image gradient magnitudes and orientations are sampled around the keypoint location, a Gaussian weighting function with σ equal to one half the width of the descriptor window is used to assign a weight to the magnitude of each sample point, orientation histograms over 4x4 regions are created and a descriptor vector is created by concatenating all orientation histogram entries. This descriptor vector is normalised. Finally, a SIFT key point descriptor vector is obtained that contains 128 values.

- Speeded Up Robust Features (SURF)

SURF algorithm is also known as an approximate version of SIFT. Main idea of the algorithm is similar, however in SURF authors drew attention to the performance and applied algorithm optimalisations. As presented by authors, the algorithm outperforms SIFT in terms of the quality and performance. Scale pyramid is not constructed as in SIFT, instead different filter sizes (octaves) are used to achieve the same purpose (the scale space is analysed by up-scaling the filter size rather than iteratively reducing the image size[13]). Main algorithm steps are:

1. 2 Aproximate Laplacian of Gaussian images are created by using a box filter and integral images: $I_{int}(x,y) = \sum_{i}^{x}\sum_{j}^{y} I(i,j)$. Integral images are optimisations for computations of intensities of any rectangle from the original image.
2. Key points are detected by applying Hessian determinants. Hessian matrix eigenvectors construct an orthogonal basis that shows direction of the curve of the image, local extremum can be detected easily when product of eigenvalues is positive. Maxima are interpolated in scale and image space.
3. To get rotation invariant key point descriptor, a Haar wavelet responses are calculated in x and y direction (invariant to light intensity changes), in a circular neighborhood of radius $6s$ (s is the scale of the detected key point). Points are weighted using a Gaussian of $2.5s$ and dominant orientation (vector) is found: sum of all responses within a sliding orientation window that covers angle of $\frac{\pi}{3}$ is calculated, and a new vector for the sliding window is obtained by summing horizontal and vertical responses.

4. Descriptor of the key point is generated. A region is constructed from a square of size $20s$ that is centered on each key point, orientation is the same as previous step vector. Region is divided into 4x4 square sub-regions, a feature vector is calculated for each region: haar wavelets are calculated for sub-region, and wavelet responses are summed: $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$ with respect to the detected orientation. Finally, a SURF key point descriptor vector is obtained that contains 64 values.

- Binary Robust Invariant Scalable Keypoints (BRISK)

BRISK is a novel algorithm presented recently. As authors state the quality of key point detection is compared to top state-of-the-art key detector SURF, but needs less computation time. This is achieved by detecting key points in octave layers of image scale pyramid, but also in layers in-between in continuous domain via quadratic function fitting. Brief description of the BRISK method [6]:

1. Octaves in scale pyramid are created by half-sampling original image, in addition intra-octaves located in-between layers are created by downsampling original image by a factor of 1.5 and successive half-sampling.
2. A FAST[14] corner detector with a mask of 9 consecutive pixels in a 16-pixel circle is applied on every layer and sub-layer (intra-octave) with the same threshold T and ROIs are detected.
3. Within detected ROIs all points are tested to meet maximum condition against its 8 neighboring FAST scores in the same layer (a corner is detected). Then, scores in layer above and below are checked and need to be lower than this detected maximum.
4. To limit complexity, a 2D quadratic function is fit to the 3x3 patch surrounding the pixel and three sub-pixel maxima are determined (from layer of the keypoint, one above and one below). Maxima are interpolated using 1D quadratic function along the scale space and local maximum for score and scale is found. Image coordinates are interpolated between patches in layers to the scale found.
5. Descriptor of the key point is generated. A sampling pattern is defined: these are N locations equally spaced on circles concentric with the keypoint[6]. A Gaussian smoothing is applied with σ_i proportional to distance between points and respective circle. Two subsets are created: short-distance pairs and long-distance pairs. A local gradient is calculated for long-distance pairs, gradients are summed and feature orientation α is found, then a descriptor is formed from short-distance pairs that are taken from a sampling pattern rotated using that orientation: a bit-vector is constructed by performing comparison of all short-distance pairs:

$$\forall(p_i^{\alpha}, p_j^{\alpha}) \in S,\ b = \begin{cases} 1, I(p_j^{\alpha}, \sigma_j) > I(p_i^{\alpha}, \sigma_i) \\ 0, otherwise \end{cases}$$

4.2 Product detection and recognition

First, original product images database is created. From all images from the products database keypoints are detected using one of features detector described in section 4.1. Keypoints feature descriptors associated to each product image from the database are stored, these are reference vectors of descriptors used for products recognition. Thus, for each product or brand logo image, a vector P_k of features descriptors is created. Note, that this vector can have various sizes, depending on complexity of the image.

When a new image is acquired by a mobile camera, first the keypoints are searched using the same detector as used for the database creation. When a set of image keypoints descriptors W_{all} is obtained, the image is scanned using a sliding window, which size is selected iteratively, starting from a small size and iteratively doubled. Image is divided into d equal-sized areas. Starting value of the divider parameter d can be tweaked based on the expected accuracy of the recognition[3]. It is important to note, that image division is needed to correctly associate training image keypoints with these from a window. If a window overlaps two identical images that have the same keypoints, but in different places, then the calculation of proper homography is problematic. Thus, that parameter has a key role in a definition of a minimal size of the training image visible in the captured image.

For each window size the image is scanned using a sliding window: keypoints inside that window are extracted, detection is performed and finally recognised keypoints are removed from W_{all}. The sliding window is moved from top-left position to bottom-right in steps. The step size is set to a half of the window size (independently: half-vertical and half-horizontal). When sliding is finished the window size is doubled to include bigger area and so on. Process is stopped when previous window size encompassed the original image.

During scanning of sliding window only a part of keypoints from the whole image are taken into consideration, such that coordinates of the keypoints are inside current area of the window. Detection of products and recognition is performed using a technique called keypoint matching [15], but only keypoints detected in a current

[3] for example 1/4 of original image size, but also one can get good results with 1/8 or even 1/2.

area of the sliding window are considered (a set $W_{i,j}$). A nearest neighbour kNN search is performed on each product keypoints descriptors vector (P_k) and window's keypoints set ($W_{i,j}$) with[4] $K = 2$. Found pairs are filtered to find good matches using technique described in [17]: first, the minimum distance (*min*) is found from all matches, and then all distances that are bigger than[5] $2 * min$ are discarded. If the amount of keypoints in a set containing found matches is less than 4 (thus, at least four corners), then operation is repeated for next product vector (P_{k+1}), in other case the product is detected, marked by product code and added to result set R. Finally a homography is found using a well-known RANSAC [18] algorithm using pairs of keypoint matches and then perspective matrix transformation of vectors is performed. If the transformed polygon is not convex, then the object is not added to the result set R. The image is scanned using a window to correctly find all copies of the product packages. If the kNN-2 would be performed on the whole image, only one copy of the product package would be detected[6]. When product is detected in the original image, all the keypoints that are inside a product's box in the transformed perspective are removed from the original set W_{all} of keypoints descriptors. Still, this could lead to over-detection. Detected areas have to be checked for such situation and repeated detections filtered out, this is described in section 4.3.

Keypoint matching can be improved for example by supervised learning [19]. In this study it was not necessary, as the results for the presented purpose are satisfactory.

4.3 Over-detection

With the procedure described above it is possible that some products recognised in a resulting set R will be repeated. To solve that, product box areas that are overlapping by more than 75% are examined. Note, that the resulting transformed box is in fact a trapeze (a polygon). Combinations of two polygons from the set R are considered for removal. To find overlapping parts of two polygons a Yu-Peng Clipper [20] algorithm is used to calculate intersection of the polygons. If the intersection area size is bigger than a mean area size of two candidate polygons, then the smaller polygon is removed from the resulting set R. Thus, at the end all over-detected polygons are

[4] The k=2 in kNN is suggested by J. Beis and D. Lowe in their Best-Bin-First (BBF) algorithm [16]

[5] This parameter was set empirically. In the literature different parameters for this distance can be found.

[6] Actually it's possible to construct an algorithm that will avoid a sliding window. Author is working on a solution of this problem

removed. Yu-Peng Clipper algorithm is not perfect and in very rare scenarios it can fail [20], but the speed gained in a whole procedure is crucial and that can be accepted in a mobile environment.

A flowchart diagram of the whole algorithm is presented in Figure 1 for one image class variant. Different training image classes are reocognised iteratively, for each training image a movement of sliding window is repeated.

5. Selecting the keypoint detector algorithm

It is very important to select proper keypoint detector algorithm depending on run time and image conditions. Considering, that the whole procedure will be run in a mobile environment with limited resources, the selected algorithm must be robust. Thus, the speed and run time of the algorithm is favored over the accuracy, but from the other hand high accuracy also must be achieved. The keypoint detector is most crucial part of the whole procedure, as it's the most computation exhaustive part. Other crucial factor is keypoints descriptors comparator. For SIFT and SURF algorithms it's a kNN algorithm (see section 4.2), however for the BRISK algorithm the simple brute-force with hamming distance algorithm was used.

5.1 Methods and results

To select the keypoint detector algorithm, a database of images was created. Author captured 20 good defined covers and logos of different medicines using a mobile camera, that was marked as a learning set. In addition real-life images were captured (test images). These images presented many different medicines in different conditions (rotated, scaled and in different lighting conditions). On one test image up to 20 different medicine packages were visible, some packages also repeated. In addition, some images were not consisting any image from the learning set. Such database was evaluated using algorithms described in previous sections, but in addition tests with additional image filtering were performed (in particular, histogram normalisation).

The first test was performed as follows: on the original learning image a keypoint detector algorithm was run, keypoints descriptors were stored in a resulting set R_{learn} together with their position (x,y) on the image. That image was further processed and altered: scale was changed to a factor of $(0.1, 10.0)$ with a step of 0.1, image was rotated and additional black frame was added around the image (rotated by a step of 15° from 0° to 360°), a gaussian random noise was applied ($I_{test}(x,y) = I_{orig}(x,y) + random$) from 10 to 300 with a step of 10, and lightness was altered ($I_{test}(x,y) = I_{orig}(x,y) + lightness$) from -50 to 50 with a step of 5. After applying changes to

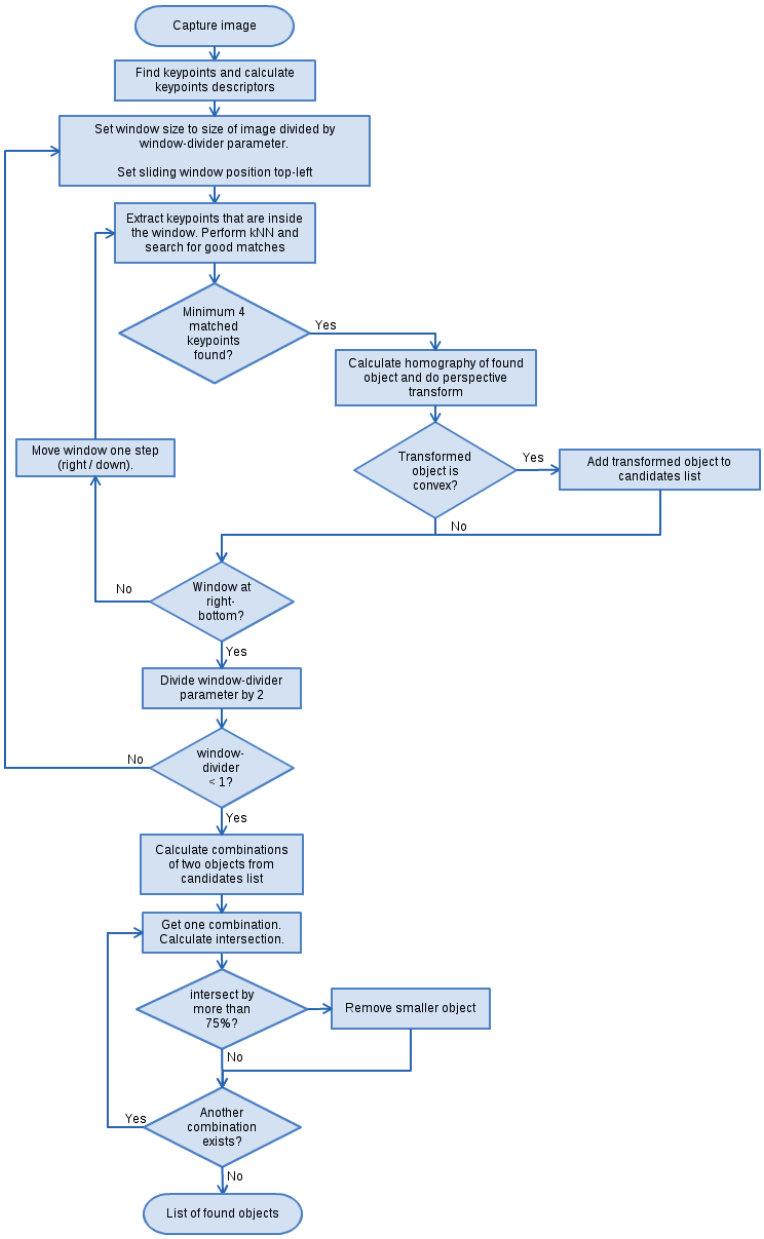

Figure 1. A flowchart diagram presenting the whole procedure from sections 4.2 and 4.3 for one image class.

the image, whole procedure described in section 4.2 was performed: on a new image keypoints were detected, keypoints pairs were found and homography was calculated. Having a new homography, the image was transformed back to origins: perspective of altered image was transformed by inverse of homography calculated. In overall that procedure moved all keypoints from altered image to positions that correspond with original learning image. For a perfect keypoints detector, all keypoints (K_{all}) from original image should have exactly the same positions as the transformed-altered keypoints. Thus, one can calculate a *ratio* how many pairs of keypoints (from original image and altered image) are exactly positioned. If distance of two keypoints from one pair was higher than 5 pixels, then such pair was discarded (keypoints do not „fit"), number of correctly positioned keypoints K_{fit} was increased otherwise. To calculate the *ratio*, a number of keypoints K_{fit} is divided by a number of all keypoints K_{all}.

On each image from original learning data set a test procedure was run, result *ratios* and *times* needed to perform the test were recorded. Then, a mean, standard deviation, max and min was calculated from recorded ratios of all images for each parameter change.

Results are shown on Figures 2, 3, 4 and 5. Lines are added to increase readability, the ratio was tested only in certain data points.

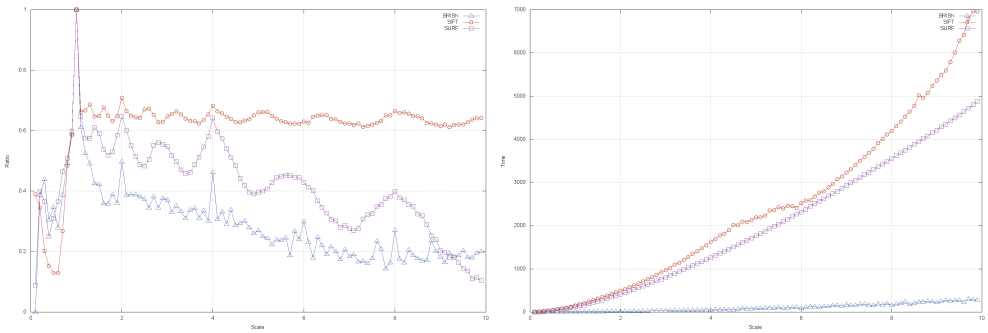

Figure 2. Mean Correct Keypoints Ratio (on the left) and algorithm run time in milliseconds (right) after changing scale (from 0.2 to 10, step of 0.1).

Another, global check was performed. In this test, all parameters (ratio, scale, noise, intensity) were changed iteratively, and all combinations of these parameters values for all images were checked. For each combination a result ratio and time was obtained. Comparison of the algorithms are presented in Table 1.

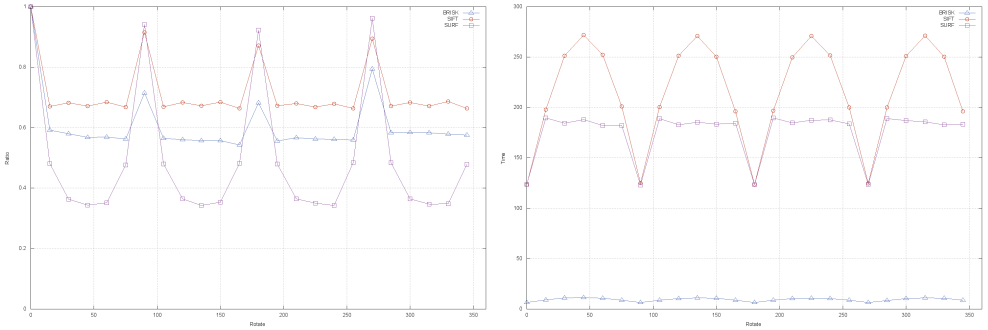

Figure 3. Mean Correct Keypoints Ratio (on the left) and algorithm run time in milliseconds (right) after applying rotation.

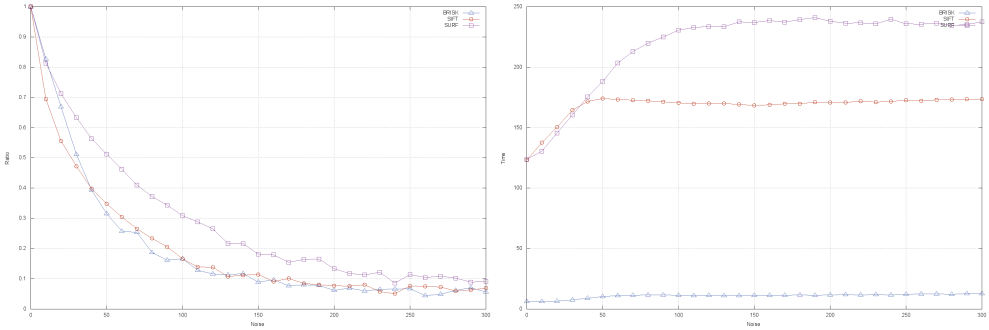

Figure 4. Mean Correct Keypoints Ratio (on the left) and algorithm run time in milliseconds (right) after applying gaussian noise.

The mean ratios are very small due to the fact that it's a comprehensive test that includes all possible image amendments, that also included a huge noise and almost zero-lightness for hundreds of tests - thus giving a small result mean ratios (that amended images are not readable also by humans). These are not real-life scenarios and were provided here only for artificial comparison of the algorithms, but surprisingly some algorithms overperformed others even in such low-quality images. The key in this table is comparison of algorithms performance in terms of speed, but also subtle difference in recognition ratios can be spotted.

From the calculated results the BRISK algorithm outperforms SIFT and SURF in terms of computational speed. The difference is huge, however the cost of computations has been replaced by lowest detection ratio. Nevertheless, the ratio is still satisfactory, and it is lower only by a fraction from the best algorithm in this field

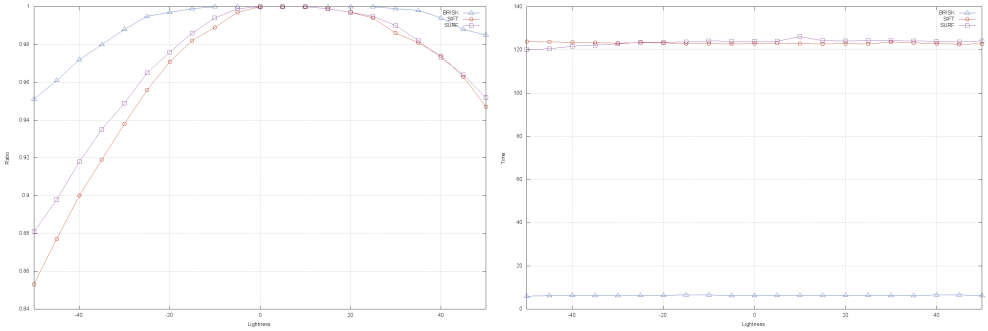

Figure 5. Mean Correct Keypoints Ratio (on the left) and algorithm run time in milliseconds (right) after amending pixels intensity.

(SIFT). Considering, that the overall procedure matches hundreds keypoints at once, and in fact only few of them are necessary to properly recognise the product, then it is obvious that the BRISK algorithm is a choice for the problem described in this paper. Not only, the robustness of BRISK allows it to be run in a mobile environment, but also good keypoints matching ratios are satisfactory. If more accuracy is needed, the SURF algorithm also performs in acceptable speed.

Table 1. Comparison of keypoint detector and matching algorithms. Table shows ratio of correct key points (the higher the better), and calculations time (the lower the better).

Algorithm	Ratio				Time (in ms)			
	mean	σ	*min*	*max*	*mean*	σ	*min*	*max*
BRISK	0.1904	0.1386	0.02	1.0	22.4	39.0	1.1	239.7
SIFT	0.2297	0.1709	0.02	1.0	744.2	1188.9	5.9	4856.3
SURF	0.2173	0.1332	0.02	1.0	429.9	639.6	3.2	2710.5

Examples of the final result of whole recognition procedure is shown in Figure 6. The reference, learning image is shown on the picture below the testing image. The lower detected box in testing image is slightly different, it's height is bigger than original learning image.

Complexity of the whole recognition procedure mainly depends on the amount of detected keypoints in the captured image. If the image is complex, the more keypoints are detected. Bigger amounts of keypoints cause a reduction of recognition speed, due to the fact that all these points have to be compared with all keypoints from training images. Also, the amount of training images is a key factor of the reco-

gnition time for the same reason. Other parameter that counts up to the recognition time is image divider (d) value. Example values of the recognition times of real-life images, running the whole procedure on iPad[7] mobile device and SURF keypoints detector, are presented in Table 2.

The memory footprint however is not an issue in this scenario. Amount of keypoints is not big, for real-life images this does not exceed several kilobytes in size, thus considering size of nowadays mobile phones memory it is just a fraction that can be omitted.

Nevertheless, further study is needed to correctly define the impact of these factors on the whole recognition procedure times.

Table 2. Example real-life images recognition times in *ms* depending on image complexity (# of keypoints detected) and startup image-divider d, using SURF keypoint detector, running on iPad device. The amount of images in training database is 7.

image resolution	# of keypoints	$d = 1$	$d = 2$	$d = 4$	$d = 8$
1000x800	1,487	1,074	3,961	9,966	23,987
1000x865	1,662	2,009	4,006	8,993	22,929
750x759	1,832	1,991	4,003	10,035	27,288
1000x800	1,868	2,025	4,963	10,987	28,044

6. Conclusions

The problem of proper recognition and counting products or visible branding logos in user surroundings can be solved by the algorithm from section 4.2. Shapes, representing branding logos, are detected using SURF or BRISK algorithm with very good rate, it's low computation time allows the user to run it on a mobile device, such as mobile phone. Proper and fast keypoints matching adds for the user possibility to capture the image in different angles and light intensities, without any calibration. However, it is still necessary to perform further study of the performance of this algorithm in a real-life scenarios and images.

Acknowledgements

This work was supported by the MB/WI/3/2012 and S/WI/1/2013. Author would like to thank QBurst Poland for help in the problem definition.

[7] machine string is *iPad3,4*

Figure 6. Example recognition marked by gray lines (top) of the reference image (bottom). Both two boxes were detected on the testing image.

References

[1] M. Skoczylas, W. Rakowski, R. Cherubini, and S. Gerardi. Unstained viable cell recognition in phase-contrast microscopy. *Opto-Electronics Review*, 19(3):307–319, 2011.

[2] D. Ding, J. Yoon, and C. Lee. Traffic sign detection and identification using SURF algorithm and GPGPU. In *SoC Design Conference (ISOCC), 2012 International*, pages 506–508, 2012.

[3] J. Pan, W. Chen, and W. Peng. A new moving objects detection method based on improved SURF algorithm. In *Control and Decision Conference (CCDC), 2013 25th Chinese*, pages 901–906, 2013.

[4] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[6] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: Binary Robust invariant scalable keypoints. *Computer Vision, IEEE International Conference on*, 0:2548–2555, 2011.

[7] H. Li, T. Xu, J. Li, and L. Zhang. Face recognition based on improved surf. *2013 Third International Conference on Intelligent System Design and Engineering Applications (ISDEA)*, pages 755 – 758, 2013.

[8] B. Besbes, A. Apatean, A. Rogozan, and A. Bensrhair. Combining surf-based local and global features for road obstacle recognition in far infrared images. *2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1869 – 1874, 2010.

[9] Z. Zheng, H. Zhang, B. Wang, and Z. Gao. Robust traffic sign recognition and tracking for advanced driver assistance systems. *2012 15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 704 – 709, 2012.

[10] Z. Wang, H. Xiao, W. He, F. Wen, and K. Yuan. Real-time sift-based object recognition system. *2013 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 1361 – 1366, 2013.

[11] S. Yu, S. Zheng, H. Yang, and L. Liang. Vehicle logo recognition based on bag-of-words. *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 353 – 358, 2013.

[12] P. Lukashevich, B. Zalesky, and S. Ablameyko. Medical image registration based on surf detector. *Pattern Recognit. Image Anal.*, 21(3):519–521, September 2011.

[13] E. Oyallon and J. Rabin. An analysis and implementation of the SURF method, and its comparison to SIFT. *Image Processing On Line*, 2013.

[14] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *In European Conference on Computer Vision*, pages 430–443, 2006.

[15] C. Huang and C. Chen. Keypoint matching technique and its applications, 2009.

[16] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. *In Conference on Computer Vision and Pattern Recognition, Puerto Rico*, pages 1000 – 1006, 1997.

[17] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[18] M. Fischler and C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[19] H. Sergieh, E. Egyed-Zsigmond, M. Doller, D. Coquil, J. Pinon, and H. Kosch. Improving SURF Image Matching Using Supervised Learning. In *Signal Image Technology and Internet Based Systems (SITIS), 2012 Eighth International Conference on*, pages 230–237, 2012.

[20] Y. Peng, J. Yong, W. Dong, H. Zhang, and J. Sun. Technical section: A new algorithm for boolean operations on general polygons. *Comput. Graph.*, 29(1):57–70, February 2005.

AUTOMATYCZNE ROZPOZNAWANIE WIELU MAREK W OBRAZACH NA URZĄDZENIACH MOBILNYCH

Streszczenie: Widoczność produktu na półce jest ważnym zadaniem nowoczesnych zasad marketingu. Bardzo często firmy mają umowy ze sprzedawcami, że konkretny produkt będzie widoczny na półce w określonym procencie w stosunku do innych widocznych produktów. To banalne zadanie sprawdzenia liczby produktów lub widocznych logo marki, jest wykonywane ręcznie przez biegłego rewidenta, który sprawdza, czy warunki umowy są spełnione. Nie istnieje łatwy, mobilny mechanizm pozwalający w prosty sposób policzyć zdefiniowane produkty, które są widoczne w otoczeniu użytkownika. Taki scenariusz może jednak zostać osiągnięty przy użyciu nowoczesnych telefonów komórkowych. Za pomocą kamery można uchwycić obrazy otoczenia, a następnie wykorzystać moc obliczeniową urządzenia mobilnego do wykrywania produktów na obrazach, by finalnie obliczyć ilość widocznych produktów. W tym celu detektory punktów kluczowych w obrazach (np. algorytmy SIFT, SURF lub BRISK) są wykorzystywane do tworzenia bazy danych obrazów produktów, a wyodrębnione deskryptory punktów kluczowych są przechowywane. W nowym obrazie punkty kluczowe znajdowane są przy użyciu tego samego detektora, ale aby uniknąć problem wykrywania wielu identycznych punktów kluczowych, obraz jest podzielony i analizowany stosując przesuwne okno. Punkty kluczowe znajdujące się wewnątrz okna są wyodrębniane i są rozważane jako kandydaci występujące na obrazach treningowych. Przy wystarczającej ilości potwierdzonych punktów obliczane jest przekształcenie perspektywy i jeśli wykryte rogi są prawidłowo ukształtowane to produkt jest oznaczony jako rozpoznany. W tej pracy zostanie zaprezentowany algorytm, który umożliwia w środowisku mobilnym rozpoznawanie oraz liczenie widocznych produktów w otoczeniu użytkownika.

Słowa kluczowe: monitoring produktów opakowań logo rozpoznawanie detekcja punktów kluczowych urządzenia mobilne

Artykuł zrealizowano w ramach pracy badawczej MB/WI/3/2012 oraz S/WI/1/2013.