Video-Based Content Extraction Algorithm from Bank Cards for iOS Mobile Devices

Rykhard Bohush[1] [0000-0002-6609-5810], Alexander Kurilovich[1]
and Sergey Ablameyko[2] [0000-0001-9404-1206]

[1]Polotsk State University, Blokhina st., 29, Novopolotsk, Republic of Belarus
bogushr@mail.ru, cfif921@yandex.ru
[2] Belarusian State University, Nezavisimosti avenue 4, Minsk, Republic of Belarus
ablameyko@bsu.by

Abstract. This paper proposes an algorithm for information fields detection and recognition of bank cards in video sequences, obtained from the iOS mobile device camera. For this, we use two basic steps. The first step is preprocessing for localization of symbols. The second step is symbol block recognition using OCR system. Preprocessing algorithm includes card edge detection, information fields segmentation, segments enhancement, symbols edge emphasizing and the final step is symbol block recognition using OCR system. Based on our approach and iPhone SDK frameworks, OpenCV and Tesseract library the bank card details recognition software is implemented. For experiments our database of the real static images and video sequences was used.

Keywords: Bank Card, Flexible Forms, Video Sequences, Card Detail Recognition, Mobile Device.

1 Introduction

The rapid growth of wireless technology has increased the number of mobile device users, which in turn has given pace to the fast development of e-commerce. The new type of e-commerce transactions, conducted through mobile devices is called mobile commerce, increasingly known as m-commerce. Mobile commerce takes a special place among innovative systems and gives an opportunity to manage banking account with the help of a mobile device effectively. Many payment apps or websites with bank card forms require manual entry of card details for payment transactions which always creates a monotonous and boring user experience. It is a time-consuming process, which requires attentiveness and diligence. Therefore, algorithms and software development for the recognition of bank card details for mobile devices is relevant.

Bank cards are usually issued by banks. However, designing them may be done by different partner merchants or organizations who partners up with the bank, which gives freedom to their designers. That is why it is impossible to rely on any graphic bank card characteristics. It can have any color and this color can coincide with color of the background on which it is placed. It leads to insufficient contrast at card borders and "false" borders or gaps. Glossy surface plastic is the main material of which bank cards are made. It has strong reflective characteristics and with the bright light, it gives highlights and flashings. In addition, if there is not enough light the image can be underexposed.

Bank card is a typical example of document with flexible form, so any data recognition algorithm can be applied during its processing. In [1] an approach for flexible form fields recognition is described using an example of credit card expiration date, which focused on the template search of data fields on card surface. Canny edge detector was used to obtain the rectangular areas and Radon Transform was used for line detection. This approach has a large computational complexity that could hardly be applicable on mobile devices in real time. In [2] an algorithm to detect business card in an image is presented. It is based on the Sobel edge detection method and Hough transformation, which is used to transform detected edges into parametrical form for further processing. However, it also has a large computational cost. The method [3] uses the classical Hough transform to receive a set of traversing lines and perform boundary detection at later stage. But the authors use a very intensive information suppressed technique at first, with downsizing the image to 180×100 and performing watershed transform. All the steps of this method allow document boundaries detection in every frame of a video acquired by a smartphone or a tablet in real time.

There is a description of business cards processing in [4] and there are two stages at the pre-processing step of this algorithm. At the first stage, it excludes background by crude approximation and extracts the business card; at the second stage, the connected components on the card surface are classified. Then there is a process of threshold binarization which separates text from the image background. However, this algorithm can only be applied to simple images without any background or monotone images. Moreover, background text is not taken into account and projective transformations are not made.

In [5] an approach for optical character recognition on Android platform using an example of business card is presented. Otsu adaptive binarization makes an image black-and-white. Then the connected components analysis is used in order to clear the foreground noise and merge potential characters into blobs. With the help of X- and Y -projections of the binary image, the algorithm approximately detects the text lines and characters. After detection all of the possible text lines in the document, Tesseract [6] is used for word recognition line by line.

A very similar method was used in [7]. It also uses a connected component-based approach to detect text in color image and the segmentation is done horizontally and vertically by using histogram projection approach.

Another Business Card processing method for Android with downsizing the input information coupled with classic transformations is considered in [8]. Canny edge detection for small grayscale image and probabilistic Hough line transform to find line segments in the detected edges are used. All the lines are analyzed for intersections and the largest quadrilateral is selected. After that a sliding window with fixed size was used to locate text boundary. Each bounding box is binarized independently using Otsu's method. The classical transformations are performed with the help of OpenCV library, and the OCR system is presented by Tesseract.

Automatic segmentation of business card region is performed by minimizing local-global variation energy in [9]. Based on this, so-called boundary "chain code" of the business card image region and the binary masks of both the business card and background regions are formed. Then, the algorithm fits a quadrangle on the business card region. This step will identify four corner points of the quadrangle shaped business card region. With the four corner points, they estimate the physical aspect ratio of the business card, thus rectify it to be rectangular by homography.

Using video instead of static images can improve content detection and recognition of bank cards. Because the usage of video sequences allows to process bank card image for a sufficiently large number of times, which is an advantage compared to using static images. Video sequences are less susceptible to incorrect orientation, as well as lack of brightness and glare. Using this approach we can significantly increase probability of positive recognitions.

This paper is organized as follows. In Section 2, a bank card content detection and recognition algorithm for video sequences obtained by mobile device video camera is illustrated. A software implementation description, the process of determining algorithm constants, experimental results using database of bank cards for content recognition and comparison of our approach and CardIO SDK are presented in Section 3. Finally, in Section 4, the conclusion and future work are provided.

2 Algorithm description

The proposed algorithm for Bank Cards Details Recognition in iOS Mobile Devices is a group of different modules that require the following steps described below.

A frame sequence obtained from the mobile video camera is given on the input of the rectangles detection block. This block detects all the rectangles in the frame and returns only that variant that satisfies the conditions of a bank card. The rectangular area is converted into grayscale and sent to segmentation block, in which the card image is separated into parts by the type of information fields. The next block carries out some transformations that improve contrast qualities of the input images and reduce the noise. Then the adaptive binarization coupled with morphological operations is applied to reduce the amount of information and calculations. After this, the sliding window is applied to images to accurately determine the symbol boundaries. Next, the separated and adjusted regions are sent to the OCR system block for the textual data extracting. Finally, the data evaluation block processes this information to obtain the result of card details recognition.

2.1 Card detection

The input for the algorithm is a video stream received from a video camera of mobile device. Given the characteristics of iPhone 6, the camera has an 8-megapixel CCD and dual-LED flash. The frame size can vary depending on particular device model, and can reach 3840×2160 pixels. The operating system is capable of automatically holding focus, adjusting brightness and white balance. These indicate that the image received directly from the camera will have effective parameters: depth of field, brightness, sufficient frame size for accurate perception of bank card details.

Separation of bank card image from background is carried out with algorithms based on Viola-Jones high-speed object detection method [10] and deep learning technique using the OverFeat method [11].

The main selection criterion from set of detected rectangles *R*, where the number of founded rectangular areas - *N*, is a ratio of its sides. Because, dimensions of a card sides $m_0 \times n_0$, the ratio of its sides is a constant value $\frac{n_0}{m_0}$. At the first, we represent the object of the found rectangle r_i as $R_i(m_i,n_i)$, where i=0,1,…,N-1. Then all the rectangles belong to the regions of interest if they satisfy the condition:

$$\frac{n_i}{m_i} = \frac{n_0}{m_0}. \tag{1}$$

The rectangle with the longest side n_{max} among all selected is the exact match. The size of the rectangular region $r_0(m_{max},n_{max})$ and its location relative to the whole image $p_0(x_0,y_0)$ are used to extract the card region from the original frame.

2.2 Regions of interest segmentation

Regions of interest (ROI), which contain information about the bank card number, the expiration date, the cardholder name, are being separated from the card image *I* with size $m_{max} \times n_{max}$ (further as $m_I \times n_I$). The size and location of these regions are defined by ISO/IEC7811-5.4:2018 and can be written as:

$C(x_C, y_C, n_C, m_C)$ - the bank card number;

$D(x_D, y_D, n_D, m_D)$ - the expiration date;

$E(x_E, y_E, n_E, m_E)$ - cardholder name.

Because obtained image size $m_I \times n_I$ of a card *I* can vary, it is necessary to convert *C*, *D* and *E* to forms that will be applicable for their correct extraction from *I*. This requires defining two scaling factors, by width (2) and by height (3) as:

$$c_{scale.width} = \frac{n_I}{n_0}, \tag{2}$$

$$c_{scale.height} = \frac{m_I}{m_0}, \tag{3}$$

Multiplying each parameter of these regions by the corresponding scaling factor, we can obtain the size of these regions relative to the size of input card *I*: C_I ,D_I ,E_I .

Receiving fragments of the regions C_I ,D_I ,E_I from the bank card image *I*, we extract images for card number (I_C), expiration date (I_D) and cardholder name (I_E).

A bank card number contains 16 digits distributed into 4 groups, with 4 digits each. Based on this, I_C selected in the previous step with the card number region C_I is being divided into 4 equal regions C_{I1}, C_{I2}, C_{I3}, C_{I4}:

$$C_{ij}\left(x_{CI}+\left(\frac{n_{CI}}{4}\times(j-1)\right),y_{CI},\frac{n_{CI}}{4},m_{CI}\right), \tag{4}$$

For $j=1,2,3,4$ we extract images I_{C1}, I_{C2}, I_{C3}, and I_{C4}, known as card number groups, respectively.

2.3 Image segment enhancement

Image enhancement belongs to image preprocessing methods. The first step is to convert an image to grayscale. After that, in order to increase the contrast between the symbol contours and the background, we should make some adjustments of the image.

There are several basic methods for contrast adjustment: linear contrast enhancement, histogram normalization, histogram equalization, etc.

Histogram equalization is a nonlinear process aimed to highlight image brightness in a way more suited for human visual analysis than for this task. It leads to an output image with after histogram, where all levels are equiprobable. In addition, subtle details and noise may appear on the image. All of the above are contrary to the current purpose - to increase the difference in brightness between the characters and card background.

The linear contrast enhancement method takes into account the brightness value for all pixels in the original image. This gives insufficient contrasting effect for cases with low-contrast images that have pronounced intensity areas on the histogram, and smooth attenuation regions. Therefore, it also cannot be applied.

Histogram normalization is the most effective method for our task, because it does not stretch the entire intensity range, but only the most informative part. This approach allows us not to consider all the extremal brightness values but to designate the conditions for their determination with a given accuracy; it enhances contrast effect due to loss of noise regions with rarely encountered intensities [12].

According to the current brightness Y of each image pixel, the output brightness g is defined as:

$$g_{x,y}=\frac{255\cdot\left(Y_{x,y}-Y_{\min}\right)}{Y_{\max}-Y_{\min}}, \tag{5}$$

where x, y - coordinates, Y_{min} - the minimum input brightness and Y_{max} the maximum input brightness.

To improve the processing results, it is allowed to deviate Y_{min} and Y_{max} from their immediate values, adjusting them by a certain percentage within the specified margin of error. This allows us to neglect an insignificant number of noise pixels along the edges of the histogram. Y_{minadj} and Y_{maxadj} values can be defined as:

$$Y_{\min adj}(i)=\begin{cases}Y_{\min adj}(i+1), if \sum_{j=0}^{i} f_{total}\left(Y_j\right)<T;\\ Y_i, otherwise,\end{cases} \tag{6}$$

where i=0…255.

$$Y_{\max adj}(i)=\begin{cases}Y_{\max adj}(i-1), if \sum_{j=255}^{i} f_{total}(Y_j)<T;\\ Y_i, otherwise,\end{cases} \tag{7}$$

where i=255…0. The f_{total} is a function for pixel counting in an image with a specified brightness Y_j. With an error margin of e (%) and image size of $m \times n$, the threshold T is define as:

$$T = m \cdot n \cdot e. \tag{8}$$

To determine hue of background and symbols, we calculate the average gray value Y_{aver}. The $Y_{aver} > 127$ means background color is light and symbols color is dark, and vice versa. To emphasize the edges of symbols, mathematical morphology is used [13].

If the color of the symbols is light, morphological transformation is based on the WhiteTopHat. It subtracts the open image from the original:

$$T_w(I) = I - I \circ b, \tag{9}$$

where I - original image, b - structuring element. This filter emphasizes edge details of light symbols.

The BlackTopHat is used to emphasize dark symbols. The filter subtracts the original image from the closed:

$$T_w(I) = I \bullet b - I. \tag{10}$$

The kernel b of both filters has a rectangular shape and the ($n_b \times m_b$) size, calculated as:

$$n_b = n_I \cdot 0.06; \;\; m_b = \frac{n_b}{3}, \tag{11}$$

where n_I is an image width.

2.4 Binarization

The next step is binarization of the given segments. Radical reduction in the amount of information is the main purpose of this operation. Given that symbol boundaries should be preserved, we choose the method of image binarization with an adaptive threshold, based on local region analysis.

Since the background of bank cards is very diverse, the brightness level of the background areas containing symbols can vary significantly throughout the image. This means that binarization methods with global thresholding technique are not suitable for solving this problem. Because they do not take into account the characteristics of neighborhood points, this affects the quality of the result. Binarization with an adaptive threshold based on local region analysis is described as [14]:

$$bin_{x,y} = \begin{cases} g_{\max}, \ if \ \ Y_{x,y} > T(x,y); \\ 0, \ otherwise, \end{cases} \tag{12}$$

where $Y_{x,y}$ is the brightness of $I(x, y)$, $g_{max}=255$ and $T(x, y)$ is a threshold. The value of $T(x, y)$ is considered individually for each pixel as a weighted sum (cross-correlation with a Gaussian window) of the $block\_size \times block\_size$ neighborhood of (x, y) minus c. Gaussian window coefficient matrix with size $block_{size} \times 1$ is calculated as:

$$G_i = \alpha \cdot e^{-\frac{\left(i - \frac{(block_{size}-1)}{2}\right)}{2\cdot\sigma^2}}, \tag{13}$$

$$\sigma = 0.3 \cdot \left((block_{size} - 1) \cdot 0.5 - 1 \right) + 0.8, \tag{14}$$

where $i=0...block_{size}-1$, and α is the scale factor chosen so that $\sum_i Ci = 1$ [15].

To reduce the noise and remove non-informative details from the binary image, and also to isolate the symbols from the background, morphological operations of closing and erosion are applied.

The both filter kernels b have an ellipse shape, which is better applied for OCR-B symbols (ISO/IEC 7811-6.1:2018). The kernel size is $n_b \times n_b$, where n_b - should not exceed (1/3) of the symbol thickness ($w_{symb}/3$). Otherwise, there is a probability of losing symbol outline during processing. Otherwise, there is a probability of losing symbol outline during processing.

The symbol thickness w_{symb} doesn't depend on the card fragment, and it is strictly specified for OCR-B. The font size projection (mm) to the size in pixels, relative to the full bank card region size is constant and calculated based on its full width as:

$$h = \frac{H \cdot m_0}{54}, \tag{15}$$

2.5 Symbol localization

To localize the symbol edges, the vertical sliding window method is used. The height of the window varies according to the font size, H_{cn} - card number font size (4,0mm), H_{exd} - expire date font size (2,85mm), H_{hn} - cardholder's name font size (2,65mm). Then its projections onto a height (px) relative to the size of the full card region are: h_{cn}, h_{exd}, h_{hn}. The general form of the calculation is represented:

$$w_{symb} = n_0 \cdot 0.004884. \tag{16}$$

where 54 mm is the card height (ISO/IEC 7811-6.3:2018), and m_0 is height of detected region of card (px).

Localized symbol area from the previous step, black and white symbol lists and language identifier for the card are sequentially transmitted to the Tesseract OCR system.

A white list consisting of a set of numbers from 0 to 9 and the language identifier "*eng*" are transmitted as parameters for a card number image. For an expiration date image: numbers from 0 to 9 and the symbol "/" form white list, "*eng*" is the language identifier. For a cardholder name image: "*rus*" and "*eng*" are language identifiers and a set of characters including punctuation marks, special characters and numbers form a black list. This approach will contribute to increasing the recognition rate and accelerating all the process.

3 Experimental results

3.1 Software implementation

The proposed approach is implemented using Objective-C programming language, and based on OpenCV 2.4.13 computer vision library, Tesseract OCR system and iPhone SDK frameworks, such as: CoreMedia and AVFoundation - media data management; UIKit - work with application interfaces; CoreGraphics - low-level, lightweight processing of 2D images based on the Quartz engine.

Video stream capture by the main device camera, using the AVFoundation framework, and the frame processing operations are implemented in separate threads with NSOperationQueue. The video data frequency is 30 fps (frames per second).The first thread obtains video frames and puts them in serial queue for processing if the last one is not busy. Otherwise, received frames are ignored. The second thread sequentially receives frames from the capture queue and starts processing them until the results are obtained. After the algorithm is finished the current frame is removed, the queue is freed and the processing iteration is repeated for new frame.

Tesseract OCR library also performs its calculations asynchronously in the background. After the algorithm is finished, we get the results using callback function in the main thread and display them in the user interface.

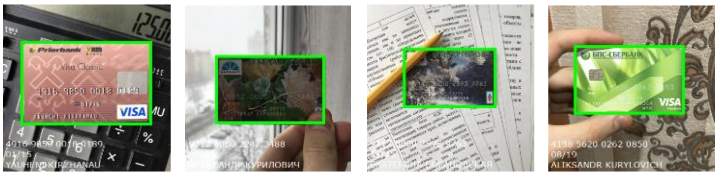
The mobile application interface is shown in Fig. 1.

Fig.1. Our mobile app interface for content recognition: 1 - the viewing area; 2 - highlighting of bank card successful detection; 3 - the data output area

The interface is equipped with an image viewing area, captured by the mobile device camera in real-time, a data output area, a mark of successful fixation on the recognition object. The capture area has a proportion of 4:3, which is the standard for a vertically oriented iPhone. The data output area contains three text fields, arranged vertically. Information recognized by the algorithm (card number, expiration date, cardholder name) is sequentially displayed therein. The card detection mark is made in the form of a bright green rectangle with a fixed thickness of borders. When a bank card position is successfully determined the mark repeats its boundaries.

3.2 Content extraction and recognition

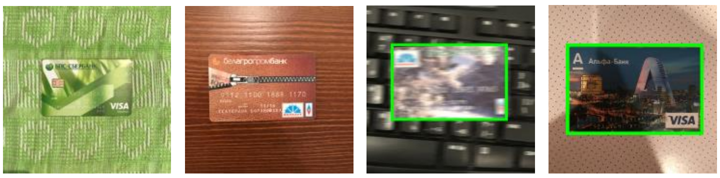
The proposed algorithm has been evaluated using data set utilizing the images and video snippets having real bank card pictorial with ground-truths and degradations. Especially for algorithm result evaluation a database of 180 static test images with characteristics and final results known in advance was formed. Fig.2 presents the results of correct card detection and recognition, (b) – (d) show that the card area is intersected by the external contour of foreign objects (b, d - fingers; c - pen). But this does not affect the detection of the card and the further recognition process.



a) b) c) d)

Fig.2. Examples of correct detection and recognition of bank cards

Fig.3 shows some error examples for card detection and recognition. Colors of card surface and background are very similar, that prevent to recognize its borders in Fig.3a and Fig.3b. The input image is blurred and fuzzy in Fig.3c. There is not enough light and the image is underexposed, furthermore color of symbols is completely the same as the background in Fig.3d.



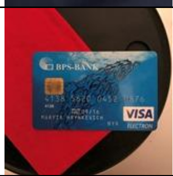
a) b) c) d)

Fig.3. Examples for bank cards detection and recognition with errors

From the experimental data it can be seen that a number of drawbacks inherent in static images are partially applicable to video sequences. For example, the completely damaged characters of the card number cannot be identified in the video as well, and the problem of matching the background also remains. Video sequences are less susceptible to problems with incorrect orientation, low light conditions and the presence of flares.

Table 1 shows partial results of the study of card data recognizing process in video sequences. The first column of the table contains examples of test video sequences frames. The second column reflects the number of frames from their total number on which the card region was successfully detected with the algorithm. The third column contains data on the number of frames in which all card details were recognized correctly, compared with the total number of frames. And the last column reflects the time from the start of the frame processing to the first correct result.

Table 1. Algorithm experimental results

Frame image	Card detected (frame count)	Symbols are recognized correctly (frame count)/total frame count	The total time processing (ms)
  	22	9 / 90	1217
	-	- / 90	-
	26	10 / 90	1221
  	23	10 / 90	1284
	26	11 / 90	1149
	31	13 / 90	1265
  	24	- / 90	-
	21	7 / 90	1344
	-	- / 90	-

The results show acceptable adaptation into different defect types such as noise, resolution and illumination changes. Our algorithm showed robust behavior in most situations and performed rather well against the comparison techniques. For example, the results presented in this paper were compared with the well-known CardIO SDK [16] and proved our method superiority in recognizing cards with Cyrillic and non-embossing characters, as can be seen from the Fig.4.

Fig.4. Content card bank extraction: a,b,c) using CardIO SDK; d,e,f) using proposed algorithm

4 Conclusion

In the paper, we have developed a fairly accurate and effective methodology for bank card information recognition. The proposed algorithm, represents the processing of frame sequence received from the mobile device camera and includes: bank card detection using the Viola-Jones high-speed object detection algorithm and the OverFeat method; ROI segmentation; improving the quality of symbols using the method of histogram normalization and morphological transformations TopHat; emphasizing the boundaries of symbols using the binarization with the adaptive threshold, performing morphological transformations on the results and searching for the most suitable region using a vertically sliding window. The Tesseract OCR engine is used to recognize characters within determined and adjusted text regions.

During the experiments, it was observed that the developed algorithm is robust to difficult conditions such as various backgrounds, light reflections and card borders, partially occluded by the user's hand, moreover, there are no false detections along with card movement. It was further noted that the performance of Tesseract OCR engine is highly dependent on the quality of the preprocessed image.

The developed preprocessing algorithm works reasonably well. It is worth noting that when we deal with symbols located on a background with low contrast, the qualitative characteristics of the algorithm are reduced. In further work, the algorithm needs to improve the quality characteristics for a low-contrast card background.

References

1. Sheshkus, A., Nikolaev, D., Ingacheva, A., Skoryukina , N.: Approach to the recognition of flexible forms on the example of the credit card date recognition. In: Proceedings SPIE

9875, Eighth International Conference on Machine Vision (ICMV 2015), pp. 83-88. Barcelona, (2015)
2. Christian, T., Gustavsson, D.: Content recognition of business cards. In: Summer Project, IT University of Copenhagen, Copenhagen (2007)
3. Puybareau, E., Geraud, T. Real-Time Document Detection in Smartphone Videos. In: Proceedings of the 24th IEEE International Conference on Image Processing (ICIP), pp. 1498-1502. France (2018)
4. Mollah, A., Basu, S., Das, N., Sarkar, R., Nasipuri, M., Kundu, M: Text Region Extraction from Business Card Images for Mobile Devices. In:Proceedings of International Conference on Information Technology and Business Intelligence (ITBI-09), pp.227-235. Nagpur, (2009)
5. Bhaskar, S., Lavassar, N., Green, S.: Implementing Optical Character Recognition on the Android Operating System for Business Cards. In: IEEE 2010, EE 368 Digital Image Processing, pp 1-5. (2010)
6. Smith, R.: An Overview of the Tesseract OCR Engine. In: Proceedings of Ninth International Conference on Document Analysis and Recognition (ICDAR), pp.629-633.Curitiba, Paraná (2007)
7. Ryan, M. , Hanafiah, N.: An Examination of Character Recognition on ID card using Template Matching Approach. In: International Conference on Computer Science and Computational Intelligence (ICCSCI 2015), pp.520–529. Jakarta (2015)
8. Sharma, P., Fujii, K.: Automatic Contact Importer from Business Cards for Android. Stanford University, Stanford (2013)
9. Hua, G., Liu, Z., Zhang, Z. , Wu, Y.: Automatic business card scanning with a camera. In: Processing of International Conference on Image Processing, pp. 373-376.Atlanta (2006)
10. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 511-519. Kauai (2001)
11. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun Y.: OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In: Processing of International Conference on Learning Representations (ICLR), pp. 1055-1061. Banff, Canada (2014)
12. Bovyrin, A., Druzhkov, P. , Yerukhimov, V., Zolotykh, N., Kustikova, V., Lysenkov, I., Meerov, I. , Pisarevsky, V., Polovinkin, A., Sysoev, A. Development of multimedia applications using OpenCV and IPP libraries. Course of the Intel Academy, Nizhny Novgorod, (2017)
13. Gonzalez, R., Woods. R.: Digital Image Processing. 4th edn. Pearson Education. United States (2017)
14. Singh, R., Roy, S., Singh, I., Sinam, T., Singh. M.: A New Local Adaptive Thresholding Technique in Binarization. IJCSI International Journal of Computer Science Issues 8 (6), 271-277 (2011)
15. Bradski, G., Kaehler. A.: Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. 2nd edn. O'Reilly Media. Sebastopol, California (2016)
16. CardIO official site, https://www.card.io, last accessed 2019/01/24

Video-Based Content Extraction Algorithm from Bank Cards for iOS Mobile Devices

Chapter · November 2019

DOI: 10.1007/978-3-030-35430-5_15

Rykhard Bohush
Polotsk State University

Sergey Ablameyko
Belarusian State University