UDC 004.056.55

SIGNAL PROTOCOL FOR END-TO-END ENCRYPTION

ARTSIOM LIAVITSKI, TATYANA RUDKOVA
Polotsk State University, Belarus

Many people merrily go about their life without worrying at all about encryption. Posting photos on Facebook and Instagram, Snapchat, and sharing private information over a multitude of messaging apps seems so natural to most millennials! Why should we even care that most of these apps do not implement end-to-end encryption? What is end-to-end encryption, and how does it work?

Implementing end-to-end encryption in a messaging service means that the contents of any given message are only available to you (the sender) and your friend (the intended recipient). Without E2EE, your message may be encrypted while it's being transmitted to the server, but the server might be able to read it. For example, some service providers might do this to generate ads that are more specific to a user.

With E2EE, your message is encrypted at all times as it makes its way through any possible intermediaries. No one except the intended recipient has the key to decrypt it. With a good E2EE protocol, neither intermediaries (messaging app server, database), nor anyone with malicious intents would be able to read the messages you send. With the amount of sensitive information, we might be sharing via text/instant messages, this is becoming an important issue. Signal - target for analyzing, a messaging app by Open Whisper Systems

The first step in establishing an end-to-end encrypted connection between two users using Open Whisper System's Signal Protocol is generating a set of long-term identity key pair, medium-term signed prekey pair, and several ephemeral prekey pairs. These keys are generated on the client side and stored locally somewhere secure. The second step involves packaging all of the public keys and registration ID into an object (known as the "key bundle") and registering it with a Key Distribution Center. In order for USER1 to send messages to USER2, USER1 must know and have access to USER2's registration ID and public keys to start a session. Thus, USER1 must first generate her own keys and register herself with the key distribution center and request USER2's key bundle.

Once a session is established, USER1 can start sending messages to USER2. This process relies heavily on the X3DH key agreement and is what gives Signal Protocol the ability to provide forward secrecy and cryptographic deniability. This also has an additional benefit of asynchronicity and thus the ability of sending messages while being offline. While the session is active, USER1 encrypts and sends messages to USER2 using the master shared secret and USER2's ephemeral keys. This step creates a root key, a corresponding chain key, and a message chain. These are critical for maintaining forward secrecy and privacy. Thus, each message that is sent leads to the creation of a new set of one-time session (ephemeral) keys that are then used to encrypt/decrypt any future messages.

On a higher level, the Signal Protocol is a security library on steroids. Despite its novelty and growing importance, there has been few formal analyses of this protocol, whilst it has been a driving force in the world of cybersecurity.

The Signal Protocol amalgamates the Extended Triple Diffie-Hellman (X3DH) key agreement protocol, Double Ratchet algorithm, pre-keys, and uses Curve25519, AES-256, and HMAC-SHA256 as cryptographic primitives. These are all well-established, low-level cryptographic algorithms that are frequently used to build computer security systems.

X3DH kicks things off, by generating all the necessary keys between two parties to communicate. It establishes the crucial shared secret key between the two parties who mutually authenticate each other based on their public key pairs. X3DH also allows for key exchange to occur where one party is "offline", and will instead exchange it through a third-party server [1].

This is used as part of a cryptographic protocol to provide E2EE based on a shared secret key derived from X3DH. Once both parties agree on a shared secret key via X3DH, parties can then use the Double Ratchet Algorithm to send and receive encrypted messages.

Double Ratchet Algorithm is used as part of a cryptographic protocol to provide E2EE based on a shared secret key derived from X3DH. Once both parties agree on a shared secret key via X3DH, parties can then use the Double Ratchet Algorithm to send and receive encrypted messages [2].

The key exchange from X3DH outputs a master secret, which in turn is used to derive two symmetric keys: "root key" and "sending chain key". As messages are being sent and received, these keys that are attached

Primary equipment lines have a greater contrast than lines of secondary equipment. That allows operator to focused on most important parts of power system.

On the left of the screen there is a unit tree list. It is used for navigation purposes. Every power station, substation and switchgears are presented here hierarchically to allow operator to be aware of which point of the power supply system is shown in the center of the screen now so that he could jump to any other as quickly as possible.

At the bottom of the screen, the events box is displayed. It displays messages about controlled object state. There are three types of messages: notification message, normal state change messages and alert messages. Notification message notify operator about normal usual events in the power system, such as closing substation door or receiving acceptable value of the controlled parameter. Normal state change messages indicate normal changes in power system and provide feedback to operator when he changes power system state. Alert messages signal an accident for example equipment failure. Each massage type has its own color: light green for notification message, blue for normal state change messages and red for alert messages. This color palette allows operator to distinguish messages by priority instantly and focus on the most important.

Adherence to the principles stated above allow us to create GUI for industrial software that is easy and even fun to use. Easiness and fun are important because they tangibly affect the operator's ability to control the process and respond quickly when things go awry. And over time, the engagement and performance of well-designed systems provide positive returns in decision-making speed and accuracy, plant productivity, and reduced training time.

REFERENCES

1. Maeda, J. Design in Tech Report [Electronic resource] / J. Maeda. – Kleiner perkins caufield & byers, 2015. – Mode of access: http://www.kpcb.com/blog/design-in-tech-report-2015. – Date of access: 02.08.2017.
2. Wang, L. Modern Industrial Automation Software Design / L. Wanga, K. Chen Tan. – IEEE Press, 2006. – 349 p.
3. Galitz, O. Wilbert.The Essential Guide to User Interface Design / Wilbert O. Galitz. – 2nd edition. – Wiley Computer Publishing, 2002. – 786 p.

• **Layout:** Layout gives operator a frame of reference for how they should interact with an environment. Layout refers to how buttons, diagrams, and other elements are laid out on screen in an organized fashion.

• **Navigation:** every ADCS software needs well-designed navigation to give users access to critical information quickly and intuitively. Basic rules include:

o A hierarchy should be consistent
o The navigation bar should be in the same place on all screens
o A process overview should provide quick access to any part of the system

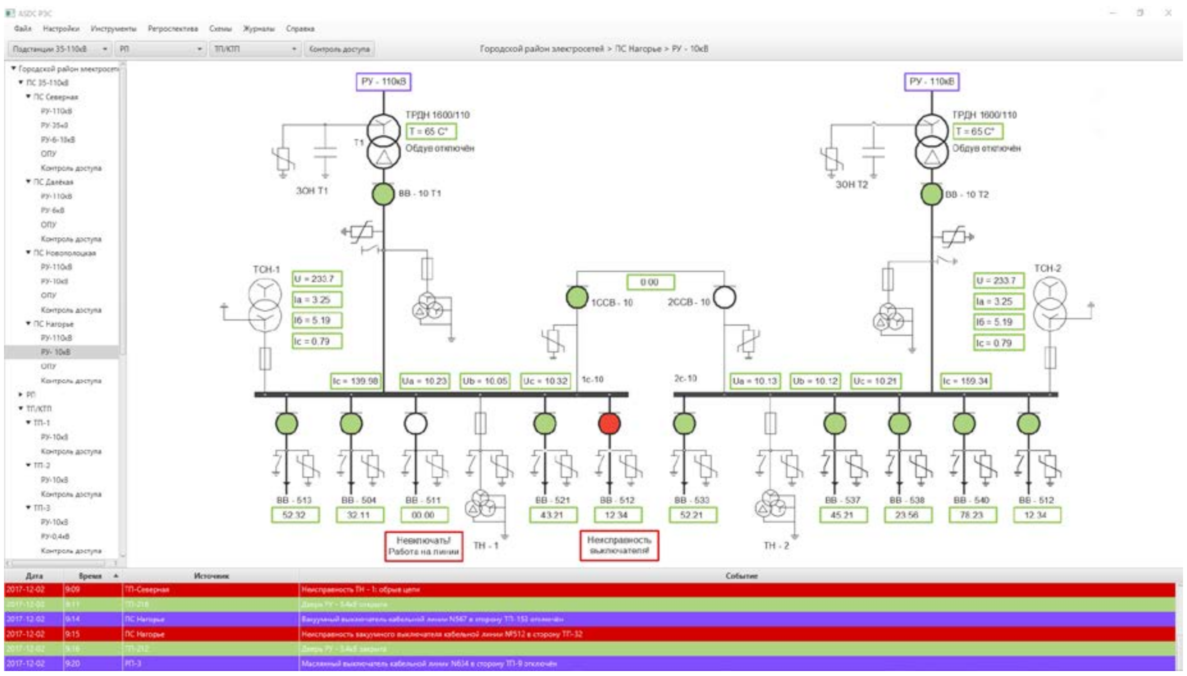
Ideally, with a well thought-out, consistent navigation system, someone without training but familiar with the plant should be able to navigate to any part of the ADCS without heading down dead ends.

• **Typography:** Typography refers to the use of fonts, font sizes, and font styling to convey meaning. Use of typography can be used to emphasize and de-emphasize content – such as by increasing contrast and boldness for important alerts, displaying hints with plain text, and using certain heights and colors for headings and subheadings.

• **Iconography and imagery:** Right image or icon could convey more information than a thousand words. However, large images are not recommended in industrial systems because screen real estate is precious and loading times can reduce speed. Instead, intuitive icons and shapes are preferred. A good icon should be simple, easy to read at different sizes, and clear in meaning.

• **Feedback:** To reach effective communication between operator and ADCS interface, it's important that the operator knows if the program understood him. Operator usually cannot see the plant and therefore needs to receive feedback that what he did had an effect.

On Fig 1. below graphical user interface of the system being developed is shown.



Дата	Время	Источник	Событие
2017-12-02	9:09	ТП-Северная	Неисправность ТН - 1: обрыв цепи
2017-12-02	9:11	ТП-218	Дверь РУ - 0,4кВ открыта
2017-12-02	9:14	ПС Нагорье	Вакуумный выключатель кабельной линии N567 в сторону ТП-153 отключён
2017-12-02	9:15	ПС Нагорье	Неисправность вакуумного выключателя кабельной линии №512 в сторону ТП-32
2017-12-02	9:16	ТП-212	Дверь РУ - 0,4кВ закрыта
2017-12-02	9:20	РП-3	Масляный выключатель кабельной линии N634 в сторону ТП-9 отключён

Fig. 1. Graphical user interface of the system being developed

GUI of ADCS consist of several parts: mnemonic diagram, units list, events box. Mnemonic diagram in the center of the screen is the main part of any ADCS or SCADA software. It displays information about the monitored process in real time and allows the operator to maintain normal system state. It was created in such a way to be much similar to power engineering drawing so that every engineer could understand it without prior training. Mnemonic diagram's color palette consist of four main colors. Light green color indicates that circuit breaker is on. Also it indicates an acceptable value of the controlled parameter. Boxes with light green border indicate that value of the controlled parameter inside the box in the normal range. If actual value deviates from the specified critical value, box borders become red. Red color indicates failures. Blue color is used to highlight jump points to other parts of mnemonic diagram. All power lines, busbars and other equipment are shown in black.

UDC 004.51

THE DEVELOPMENT OF GRAPHICAL USER INTERFACE OF AUTOMATED DISPATCH CONTROL SYSTEM

ULADZISLAU KUTAS, SERGEY SURTO
Polotsk State University, Belarus

Process control and equipment management requires operator to have a high reaction speed, rapid response capacity and ability to analyze a lot of information in a short time to make the appropriate decision. Automatic dispatch control system (ADCS) allows to simplify management process and make operator's work more efficient. ADCS provides an interface to the operator, which allows the operator to interact with process *plant in more intuitive way. In most cases the interface is a graphical user interface. In this paper, an example of graphical user interface design of ADCS of electric power distribution system is presented as well as the underlying principles of graphical user interface design.*

Automatic dispatch control system – is a control system that uses computers, networked data communications for high-level process supervisory management and other peripheral devices such as programmable logic controllers and discrete PID controllers to interface to the process plant or machinery. ADCS allows to interact with process plant through high-level interface, which is usually graphical user interface (GUI).

The reason why software giants have spent a lot of resources developing the best user interface (UI) and user experience (UX) is because they've done the research and recognize the real benefits of implementing good design [1]. Success and speed of operator's work largely depends on UX. Ill-conceived UX could lead to frequent delays and mistakes of operator. Because GUI in most cases is the main user interface of ADCS, it largely determines UX. In most cases the reason of UX ineffectiveness is the ill-considered GUI. Ultimately, providing reasonable GUI with ADCS software that makes the operators' job easier, allowing them to catch errors and resolve them immediately, minimize timewasting traps, save millions in machine damage and lost productivity cascading down the manufacturing process.

There are primarily six principles which can be used as the guidelines for user interface design [2]

1. User familiarity: the purpose of screen elements should be clear and understandable without prior training; permissible manipulations with these elements should also be intuitively understood. The user interface should not contain too much detail;
2. Consistency: A user with experience with some software can quickly adapt to any similar software. It also refers to different operations in the context of one program: the skills learned from one operation can be easily applied to other operations;
3. Minimal surprise: means that the comparable operations should incur the comparable results. If this does not happen, the users will become confused and frustrated, and even raise the doubt on the software design quality;
4. Recoverability: The system should not be sensitive to operator errors. The operator must be able to cancel any of his actions. For this purpose multiple confirmations, cancellations, backtracking, setting of control points, etc. are used;
5. User guidance: A well-designed help system is highly necessary. It should be incorporated and become a built-in component of the overall system. Also, comprehensive search and index tools should be provided to make the user query more convenient.
6. User diversity: A novice user should have a simpler interface with more tips. For an experienced user, the number of prompts should be reduced, as they interfere with work.

To implement the principles, the following components are used [3]:

• Color and contrast: Color and contrast often communicate before all other visual factors like text are taken into account. Color is used to convey instant meaning to objects, to draw attention to specific places that may need focus, and to distinguish items from each other. To create an interface that corresponds to the principles indicated above, you must select a color palette and assign each individual color with your own meaning. For example, red color means that an alarm has been triggered, or a power line is emergency shutdown. Operators must memorize the designations for each color, and therefore having a standardized set of colors will decrease confusion and speed up data interpretation.

Color contrast between lighter and darker tones can be used to make one object stand out compared to its surroundings. The greater the contrast between two colors is the more noticeable and legible it becomes.

transformations TopHat; adjusting 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 the vertically sliding window. Recognition of adjusted regions is implemented by the Tesseract library.

REFERENCES

1 Robust Real-time Object Detection Using a Boosted Cascade of Simple Features : Proceedings of the Computer Vision and Pattern Recognition Conference, 2001 / Viola, P. and Jones, M.J.
2 OverFeat: Integrated Recognition, Localization and Detection Using Convolutional Networks, December, 2013 / Sermanet, Pierre, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun.
3 Kustikova, V.D. Development of multimedia applications using OpenCV and IPP libraries : textbook / V.D. Kustikova. – Nizhny Novgorod : Nizhny Novgorod State University. N.I. Lobachevskogo, 2012.
4 Gonzalez, R. Digital image processing / R. Gonzalez, R. Woods. – M. : Technosphere, 2005. – 1072 p.
5 OpenCV documentation [Electronic resource] // Miscellaneous Image Transformations. Adaptive Threshold. – Access mode: https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html, free. – Access date: 30.11.2017.
6 OpenCV documentation [Electronic resource] // Image Filtering. getGaussianKernel. – Access mode: https://docs.opencv.org/2.4/modules /imgproc/doc/filtering.html, free. – Access date: 30.11.2017.

the size of the region of the full card are: h_{cn}, h_{exd}, h_{hn}. The general form of the calculation is represented by formula:

$$h = \frac{H \cdot m_0}{54.0} \tag{9}$$

where 54.0 mm is the height of the card according to ISO 7810 ID-1, and m_O is the height of the detected region of the card (px).

Images are sequentially transmitted to the input of the Tesseract system for recognition, with parameters: the adjusted symbol area obtained in the previous step; language.

The proposed algorithm is implemented in the Objective-C programming language, using the OpenCV 2.4.13 library (this version is the most relevant for iOS), with the help of the iPhone SDK frameworks, such as: CoreMedia and AVFoundation - media data management; UIKit (UI) - work with application interfaces; CoreGraphics (CG) - low-level, lightweight processing of 2D images based on the Quartz engine.

The mobile app interface (fig. 2) is equipped with an image viewing area, captured by a mobile device's 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/iPad. The data output area contains three vertically arranged text fields into which the information recognized by the algorithm (the bank card number, expire date, cardholder's name) is sequentially displayed. The card detection mark is made in the form of a bright green rectangle with fixed thickness of borders and is displayed on the screen by repeating its boundaries when the card position is successfully determined. The capture of the video stream from the main camera of the device is implemented in separate thread using the AVFoundation framework and the AVCaptureDeviceInput object, which is initialized by the AVCaptureDevice object in the video capture mode. A new session of AVCaptureSession is being created, with the frame size parameters (for iPhone 7, the maximum size is 3840 × 2160). Using the addInput method, a previously defined input object is added to it and via the call startRunning - the session starts. The frame data is retrieved using the didOutputSampleBuffer: from Connection method, which shows the frames in the current context of the view controller using AVCaptureVideoDateOutput. The results of the time consuming estimation of the algorithm were run on the iPhone 7 for video with different sizes of the input frame (fig. 3).

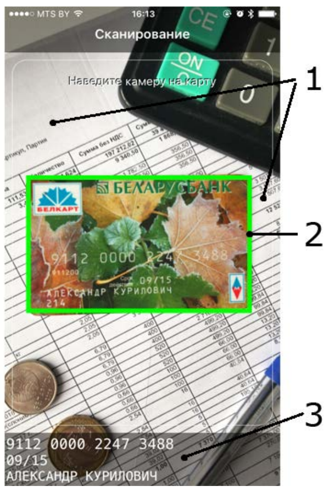


Fig. 2 - Appearance of the mobile app:
1 – the image viewing area; 2 – the mark of successful fixation of bank card;
3 – the data output area

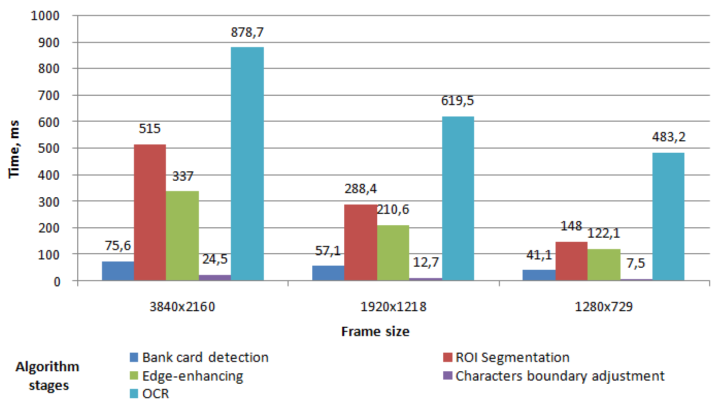


Fig. 3 - Time consuming bar graph

Conclusion

In this article, the task of recognizing information fields of bank cards is solved. The developed algorithm represents the processing of the sequence of frames received from the mobile device's 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 brightness and to designate conditions for their determination with a given accuracy, which enhances the contrast effect due to the loss of noise regions with rarely encountered intensities [3].

To determine the tone of the background and the tone of the symbols we calculate the average brightness of the image $Y_{aver.}$. If $Y_{aver.} > 127.5$, then we take the background color of the received image as light, and the color of the symbols as dark, otherwise - vice versa. Using operations of mathematical morphology, the boundaries of symbols are emphasized. If the color of the characters is light, the WhiteTopHat morphological transformation function [4] is used. For dark symbols, the BlackTopHat function is used.

The structuring element for the kernels of both filters has a rectangular shape and a size of $n_b \times m_b$, calculated by the formula (5), where n_I, the width of the image received for the morphological transformation.

$$n_b = n_I \cdot 0.06$$
$$m_b = \frac{n_b}{3} \tag{5}$$

The next step is the binarization of images with an adaptive threshold based on the analysis of the local region [5, 6]. The method converts the image in grayscale *I* to a monochrome image according to (6).

$$bin_{x,y} = \begin{cases} g_{max}, if\ Y_{x,y} > T(Y_{x,y}, block\_size, c) \\ 0, else \end{cases} \tag{6}$$

where $Y_{x,y}$ is the brightness value of the pixel with the (*x, y*) coordinates of the image *I*; $g_{max}=255$ is the maximum brightness value, $T(Y_{x,y}, block\_size, c)$ is the adaptive binarization threshold, calculated individually for each pixel; where the function *T* - is the weighted sum [5] (cross-correlation with the Gaussian window, Gaussian (7) [6]) of a block of pixels with size *block_size×block_size* adjacent to the processed pixel (with coordinates *x, y*) minus the coefficient c [5].

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

where i=0... block_size-1,

α*- is chosen in such a way that* $\sum_i G_i = 1$ (7)

To reduce noise, to remove unnecessary non-informative details from the binarized image, and roughly approximate the symbols to isolated painted blocks different from the background of the image, successive morphological operations of closing and erosion are applied to it.

Structuring element *b* of the kernels of both filters has an ellipse shape, which is better suited for processing characters which font, OCR-B, according to ISO / IEC 7811-1: 2017, has smoothed edges. The size of the kernels $n_b \times n_b$, where n_b - should not exceed the size of a third of the thickness of the symbol $\frac{w_{symb}}{3}$, otherwise there is a probability of losing the symbol outline during processing.

The thickness of the symbol w_{symb} does not depend on the type of the processed fragment of the card image, and is strictly defined for OCR-B. The projection of the font width value (mm) to the size in pixels relative to the size of the full bank card region is calculated based on its full width (8), and is a constant within the processing of the current card region.

$$w_{symb} = n_0 \cdot 0.004884 \tag{8}$$

To adjust the edges of the symbol region on the processed image, the vertical sliding window method is used. The height of the window varies from the font size of the characters. For the bank card, these are values determined by ISO/IEC 7811-1: 2017, H_{cn} - card number font height (4,0mm), H_{exd}- expire date font height (2,85mm), H_{hn}- cardholder's name font height (2,65mm). Then their projections onto a height in pixels relative to

The main criterion for selecting a contour of the bank card from the set of detected rectangles *R*, where the number of rectangular areas found- *N*, is the ratio of its sides. Since dimensions of the card sides (m_o×n_o) are defined by ISO / IEC 7811-1: 2017, the ratio of its sides is a constant value $\frac{n_o}{m_o}$. At 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 the rectangles *r* for which this condition (1) is satisfied belong to regions of interest. The exact match is the rectangle with the longest side n_{max} among all selected.

$$\frac{n_i}{m_i} = \frac{n_o}{m_o} \tag{1}$$

Dimensions of the rectangular region r_o (m_{max}, n_{max}) and its location relative to the whole image p_o (x_o, y_o) are used to extract the card region from original frame.

Regions of algorithm interest, containing information about the bank card number, the expiration date, the cardholder's name, are being separated from the card image *I* with sizes m_{max}×n_{max} (further as $m_I \times n_I$). The size and location of these regions are defined by ISO / IEC 7811-1: 2017 and can be written in the following presentation:

- region $C(x_C, y_C, n_C, m_C)$ - the bank card number;
- region $D(x_D, y_D, n_D, m_D)$ - the expiration date;
- region $E(x_E, y_E, n_E, m_E)$ - cardholder's name.

Since the sizes m_I×n_I of the obtained image of the card *I* can vary, it is necessary to convert *C*, *D* and *E* to the form that will be applicable for their correct separation from the image *I*. To this end, it is necessary to define two scaling factors: by width (2) and by height (3).

$$C_{scale.width} = \frac{n_I}{n_O}, \tag{2}$$

$$C_{scale.height} = \frac{m_I}{m_O}, \tag{3}$$

Multiplying each of regions parameters indicated above by corresponding scaling factor, we obtain the sizes of these regions relative to the size of the obtained image of the card *I*.

$$C_I(x_C \cdot c_{scale.width}, y_C \cdot c_{scale.height}, n_C \cdot c_{scale.width}, m_C \cdot c_{scale.height})$$
$$D_I(x_D \cdot c_{scale.width}, y_D \cdot c_{scale.height}, n_D \cdot c_{scale.width}, m_D \cdot c_{scale.height})$$
$$E_I(x_E \cdot c_{scale.width}, y_E \cdot c_{scale.height}, n_E \cdot c_{scale.width}, m_E \cdot c_{scale.height})$$

Separating fragments of regionsC_I, D_I, E_I from the image of the bank card *I*, we obtain images: I_C, I_D, I_E, respectively, the card number, the expiration date and the cardholder's name.

According to ISO / IEC 7810 - 2006, the bank card number contains 16 digits distributed into 4 equal groups of 4 digits each. Based on this fact, the image 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} (4).

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

$j = 1,2,3,4$ (4)

Separating the calculated regions C_{I1}, C_{I2}, C_{I3}, C_{I4} from the image I_C, we obtain respectively the images I_{C1}, I_{C2}, I_{C3}, I_{C4} - groups of the card number.

The next step is to convert the image to grayscale. After that, to increase the brightness difference between the symbol contours and the background of the images, a procedure of contrast increasing is used. The most effective for this task is method of histogram normalization, since it does not stretch the entire intensity range, but only it's most informative section. This allows us not to consider the true extremal values of the

UDC 004.932

SOFTWARE FOR SCAN BANK CARDS IN IOS MOBILE DEVICES

ALEXANDER KURILOVICH, RYKHARD BOHUSH
Polotsk State University, Belarus

In this article we propose the preprocessing algorithm for identifying information fields on the bank cards images. Bank cards details recognition software for iOS mobile devices, iPhone SDK frameworks, OpenCV and Tesseract libraries is implemented based on this algorithm. Results of the time cost for bank cards details recognition on the iPhone7 for video with different sizes of the input frame are presented.

Nowadays many M-banking apps require manual entry of bank card details (all information fields data applied on the card) into the system for payment transactions and it is not an easy task for a user. This process is time-consuming, requires attentiveness and diligence. Therefore, development of the algorithmic provision and software for recognition of bank card details for iOS mobile devices is relevant.

Offered recognition algorithm of bank card details for iOS mobile devices requires the following steps: card region detection; segmentation of the bank card image; converting colored segments to grayscale; contrast enhancement; emphasizing the boundaries of symbols using operations of mathematical morphology; adjusting the boundaries of the grouped blocks of symbols; recognition of symbol blocks by the Tesseract library. General scheme of the developed algorithm is shown in fig. 1.

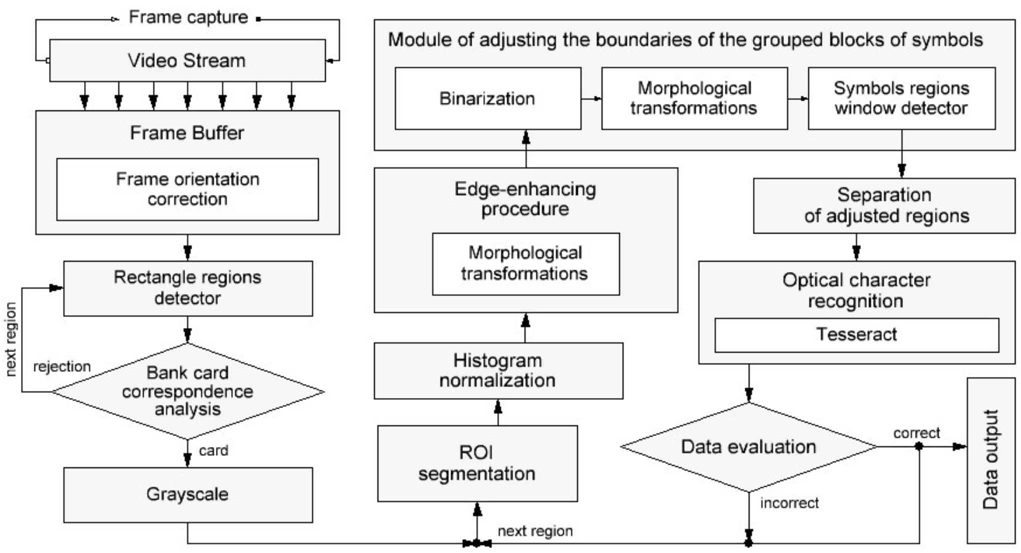

Frame capture
Video Stream
Frame Buffer
Frame orientation correction
Rectangle regions detector
next region
rejection
Bank card correspondence analysis
card
Grayscale
Module of adjusting the boundaries of the grouped blocks of symbols
Binarization
Morphological transformations
Symbols regions window detector
Edge-enhancing procedure
Morphological transformations
Histogram normalization
ROI segmentation
next region
Separation of adjusted regions
Optical character recognition
Tesseract
Data evaluation
correct
incorrect
Data output

Fig.1. General scheme of the developed algorithm

The input data of the algorithm is video stream received from the mobile device's camera. The stream is divided into frames by time and represents images in the RGB color space. Considering characteristics of the iPhone 6 (2014), the camera has an 8-megapixel CCD, dual-LED flash and autofocusing. The frame sizes can vary depending on the particular device model, and can reach 3840 × 2160 pixels. All devices younger than the above have better characteristics. The operating system (greater than iOS 2) is capable of automatically holding the focus on the subject, automatically adjusting the brightness and white balance in the photo. The above indicates that the image coming directly from the camera system will have the optimal input parameters: depth of field, brightness, sufficient frame size for accurate perception the details of the bank card.

Separation of the bank card image from the background is carried out by processing it with iOS algorithms, which are based on the Viola-Jones high-speed object detection method [1] and deep learning using the OverFeat method [2].

lower dimension (most often, two-dimensional), it is also used to solve modeling problems, forecasting, identifying sets of independent features, searching for regularities in large data sets.

Despite the possible reduction of human efforts spent on scheduling, these algorithms are rather cumbersome, and do not always provide sufficiently effective results. Thus, it is most advantageous to use algorithms to approximate the distribution of classes at the beginning of the work, and then to edit the obtained schedule manually to achieve the desired result and to take into account any additional criteria, or to use algorithms to complete the schedule partially compiled by hand.

REFERENCES

1. Береговых, Ю.В. Алгоритм составления расписания занятий / Береговых Ю.В., Васильев Б.А., Володин Н.А. ; Государственный университет информатики и искусственного интеллекта. – Донецк : Штучний інтелект, 2009 – 7 с.
2. Астахова, И.Ф. Составление расписания учебных занятий на основе генетического алгоритма / И.Ф. Астахова, А.М. Фирас ; Воронежский государственный университет. – Воронеж : Вестник ВГУ, 2013. – 7 с.
3. Anirudha, N. An Algorithm to Automatically Generate Schedule for School Lectures Using a Heuristic Approach / N. Anirudha, P.P. Manisha, G. Abhijeet // International Journal of Machine Learning and Computing. – 2012. – Vol. 2, №. 4. –4 p.