Journal of Theoretical and Applied Information Technology
31st January 2015. Vol.71 No.3
© 2005 - 2015 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FACE TRACKING FOR A SYSTEM OF COLLECTING STATISTICS ON VISITORS AND QUALITY ASSESSMENT OF ITS FUNCTIONING

A.S.SAMOYLOV, D.M.MIKHAYLOV, P.E.MININ, A.D.EGOROV
Engineering Centre of the National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Kashirskoye Highway 31, 115409, Moscow, Russian Federation
E-mail: vyndors@gmail.com , dmmikhajlov@mephi.ru , peterminin@gmail.com ,
egorovalexeyd@gmail.com

ABSTRACT

This article focuses on the development of a face tracking algorithm for a system collecting statistics about visitors. The basic method of tracking by template-matching is modified in several ways to cope with its drawbacks and achieve a stable tracking performance. In addition, the problem of objective quality assessment for tracking methods is considered, and a piece of software is described that provides a solution to the problem. Experiments conducted using this software prove the effectiveness of the proposed modifications.

Keywords: *Face Tracking, Template Matching Tracking, Template Update, Histogram Matching, Quality Assessment.*

1. INTRODUCTION

Today the collection of statistics about the visitors becomes increasingly popular in different business areas. In a competitive market, many companies want to obtain reliable statistics on the number of visitors, duration of eye contact, and other indicators. Most of these needs are initiated by advertisers who are interested to have their ads seen by the maximum number of potential buyers.

Different scientific studies mention systems that improve the effectiveness of advertising services [1-4]. For example, some models only play advertising content if there is a person in front of the media display as described in [5]. However, most counting systems are developed to measure the amount of people staying in the room.[6]. There are also powerful analytic systems for the faces of visitors, but the processing is carried out either on a remote server, or requires the use of a powerful server of its own [7; 8].

This work aims to develop a system for collecting statistics about the visitors on the basis of a small personal computer (nettop) and a webcam. The purpose of the developed system is to count the number of people who have looked at the object of interest, and to measure the time a person spent in front of the object. Some of possible applications of the system:

- Count the number of views and determine the duration of views for a booth or a poster
- Statistics for various types of advertising and shop windows
- Evaluation of attendance and audience analysis in malls

This goal is achieved by using software that detects faces in the frame and tracks their motion.

There is a variety of algorithms for object tracking, but there is still no generally accepted approach to this problem. It was necessary to analyze the methods, select one of the existing algorithms and adapt it for the specific task. Thus, the first part of the work was the creation of a stable tracking, operating in real time on a low-powered nettop.

At the moment, there are no methods or software tools to evaluate the quality of tracking. Such an assessment is carried out either by an expert or by the algorithm creator with a set of specific tests for tracking. Expert method is not protected against the human factor and does not provide a numeric evaluation. Thus the second goal of this paper was to determine the formal indicators that reflect the quality of the system, and to create a software tool that calculates such parameters for a particular algorithm. Besides assessing the quality such a software tool can be used to compare different tracking algorithms, check a new version of an algorithm to detect possible errors, as well as fine

Journal of Theoretical and Applied Information Technology
31st January 2015. Vol.71 No.3
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

tune each method and compare various tracking systems.

Thus, the task was to develop a face tracking algorithm for the system of gathering statistics on visitors in real-time with the following parameters:

- workstation with Intel Atom D525 processor or an equivalent in performance with no less than two cores;
- Web-camera Logitech HD Pro Webcam C920 or other web-camera with equivalent video quality.

It is also necessary to define a testing methodology and performance indicators of the system and introduce a software tool for evaluation of the quality of the results and tests. To achieve this goal we need to develop a software tool that would allow the operator to mark up a video, specifying the location of each face as easily and quickly as possible. The resulting reference can be compared to the results of tracking to assess its effectiveness.

2. FACE TRACKING

Viola-Jones method is a commonly accepted method for detecting faces in images [9; 10]. This algorithm is based on the representation of images in their integral form and construction of a cascade of "weak" classifiers, the so-called Haar-like features. Features used by Viola and Jones are based on the Haar wavelet [11].

To select specific Haar functions and threshold levels a machine learning method called AdaBoost [12] is used. AdaBoost combines many "weak" classifiers in order to create a "strong" one. Here "weak" means that a classifier yields correct results just a little more often than random choice. Viola and Jones combined series of AdaBoost classifiers as a series of filters, which is particularly effective for the classification of image regions. Each filter is a separate AdaBoost classifier with a relatively small number of weak classifiers. This approach was extended in 2002 by Professor Rainer Lienhart to features rotated by 45 degrees. [13]

In later studies LBP (Local binary template) features were suggested instead of Haar features [14-16]. Despite the use of integral representation of the image, the performance of LBP cascades greatly exceeds the speed of Haar cascades [15; 16].

Unfortunately, face detection can not be used for face tracking for the following reasons:

- only frontal positions are detected, while the system must monitor turned faces too;
- even frontal positions are not always detected;
- execution time of the algorithm for a single frame is still significant for face tracking in real time.

Therefore, face detection should be complemented with a specialized algorithm for tracking faces.

3. EXISTING TRACKING METHODS

Tracking is locating the object of interest in time using a camera. The problem of face tracking has been of interest for a long time and the solutions are still non-conclusive. The main difficulties of face tracking are [17-19]:

- Illumination changes. Often changes in lighting lead to face loss. Depending on the camera's position illumination can vary quite dramatically, for example, if the camera is mounted in a car and tracks the movement of the driver's head. In this paper, the problem of changing lighting should not arise, as the cameras will be installed in a room with constant artificial lights.
- Head rotation. People do not always turn to the camera frontally. Head movements are one of the main difficulties of face tracking as the image changes non-linearly when the face turns. Tracking should be sufficiently flexible and stable to track such changes.
- Facial expressions. A person's face can express many emotions (joy, surprise, anger, etc.) that change its appearance. But a face doesn’t change beyond recognition, so most tracking methods are capable of coping with this problem.
- Face occlution with other objects. Often a preson would cover a part of their face with their hand, but there is still a part of the face in the frame.
- Complexity of separation of the visual object from the background.

There are many algorithms for tracking objects, in our case, the basic tracking requirements are speed and stability. Tracking should robustly follow the person's face while it is in the frame. At the same time the algorithm should operate fast enough to maintain a certain framerate that is required to track fast moving objects. The most popular tracking algorithms are considered below.

2.1 Template Tracking

The simplest and most obvious method of tracking is searching among all possible positions of the object, calculation of similarity between the template and the image region and selection of the best value. First, we construct a matrix of similarity or difference (depending on the method used) and regions of the image at all points in a certain neighbourhood of the current position (the search area). Then a minimum or maximum value is selected from this matrix, which corresponds to the most probable position of the template. Figure 1 shows an illustration of the process.

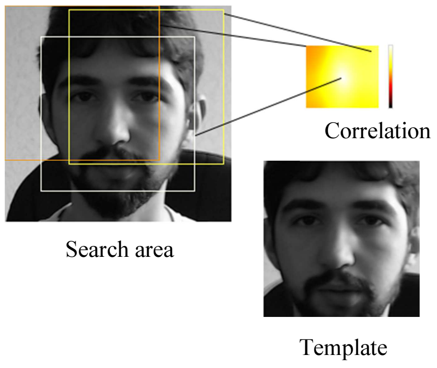

Figure 1: Visualization Of The Similarity Matrix. Brighter Points Correspond To Rregions That Match The Template Better

There are multiple methods to estimate the difference between the template and an image area. These are the most common ones [20-24]:

- Sum of absolute difference

$$R_{SAD}(x,y) = \sum_{x',y'} |T(x',y') - I(x+x',y+y')| \quad (1)$$

- Sum of squared difference

$$R_{SSD}(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2 \quad (2)$$

- Cross-correlation

$$R_{CC}(x,y) = \sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))^2 \quad (3)$$

- Normalized cross-correlation

$$R_{NCC}(x,y) = \frac{R_{CC}(x,y)}{\sqrt{\sum_{x',y'} T(x',y')^2 \sum_{x',y'} I(x+x',y+y')^2}} \quad (4)$$

Comparison methods using the sum of absolute difference and the sum of squared difference can be used non-normalized while non-normalized correlation can produce errors [21; 25]. Therefore, we shall consider only the normalized version.

Paper [21] studies the effect of noise on the accuracy of the sum of absolute differences and normalized cross-correlation. The methods showed results indistinguishable within error limits.

Normalized cross-correlation and the sum of squared differences can be calculated in the frequency domain by using the Fourier transform, which has been shown to be faster than direct calculation [22-26].

2.2 Tracking Using Optical Flow

Optical flow is a vector field of apparent motion of objects, surfaces and edges in a visual scene caused by the relative motion between the observer (an eye or a camera) and the scene.

Lucas-Kanade tracking at the time of its creation was the most high-speed solution [17], [27]. Lucas and Kanade showed a way to generalize it for the cases of translation, rotation, scaling, and other movement templates.

Subsequent years brought many modifications aimed at various improvements of the algorithm [28-30]. In order to improve the performance of the method, it was suggested to track only certain points or features of the image instead of an entire area [31]. These features are actually small areas of the image. But not any small area of the image can be called a "good" feature. It is only rational to monitor features with strong enough texture. Paper [32] proposed a selection algorithm for such features. The method based on points tracking was named after its founders of Kanade-Lucas-Tomasi. The algorithm uses the following assumptions:

- Lighting is constant
- Motion between successive frames is small
- Within a small area all points move in the same way

A pyramidal algorithm modification [33] was introduced to remove the restriction on small motion between successive frames. The method consists of using multiple scaled down copies of the image. At first the smallest image is used for "rough" estimation of the shift of a feature. The next image of the pyramid is used to refine the shift. This method allows following an object that has shifted significantly between the two frames.

2.3 Color-Based Tracking

The most popular tracking using color information is Camshift (Continuously Adaptive Mean Shift). Camshift is an extension of mean-shift algorithm for rotation and scaling cases.

The method iteratively locates the object using a color histogram, finds its size and calculates the rotation angle. HSV representation of the image is used in the original paper [34], particularly the hue channel. Using only one channel makes tracking fast enough. The saturation and value channels are thresholded in order to improve the stability of tracking as hue is not defined at low values of these parameters. Paper [27] suggests limiting both of these channels at 10%. Using the color alone eliminates problems with lighting. Moreover, this tracking proved to be effective with partial occlution of the object. The rotation angle is not determined reliably by the method.

As shown in [35], Camshift requires a high-quality camera with good color accuracy. Furthermore, the use of one channel proves insufficient for stable tracking. Saturation [36], and in some cases all three channels [37; 38] can be used for histogram calculation as an extension of the algorithm, but large histograms make the method computationally complex.

2.4 Methods Using Head or Face Models

In recent years, a dynamic field of research was the usage of head models in tracking methods. Generally, the methods of this group use two kinds of models: shape model and texture model. The models can also be divided into deformable and non-deformable ones.

A rectangle, ellipse, cylinder or ellipsoid are used as simple rigid models [17; 39]. Deformable models have more complex two-dimensional geometry reflecting face structure. Candide [40] and AAM [41] are the most common examples among them.

Methods using a model of the head or face allow to determine the position of the head, and in the case of deformable models, can adapt to a specific face. However, these methods, firstly, require model training, and secondly, have a high computational complexity. Some optimization techniques used in these methods cannot work effectively in real-time [42]. Moreover, this group of methods is sensitive to the quality of the image, and also requires fairly accurate model initialization.

3. CHOOSING THE TRACKING ALGORITHM

The tracking algorithm must meet the following requirements:

- work in real time on a PC with a low-power processor;
- stable tracking with considerable out-of-plane rotation, ideally up to 90 degrees.

Methods using a head or face model are not suitable for this task because they are very performance demanding. Camshift tracking does not provide sufficient accuracy using only the color channel, and the use of other channels makes it computationally inefficient.

Thus, Lucas-Kanade and template-matching tracking are more suitable for the task, as they are the most rapid and simple. Template tracking was the final choice, since it is more accurate and does not drift away in case of head rotation.

4. TEMPLATE-MATCHING TRACKING

This section considers adaptation of the basic template tracking algorithm to the aforementioned requirements. Normalized correlation criterion was chosen as a measure of similarity as it provides the best resistance to changes in lighting and when used in the frequency domain is computationally efficient.

Template tracking can work with the image both in grayscale and in color. In this paper we used grayscale, as processing color images requires more computation.

4.1 Initialization

A face detected by the algorithm of Viola-Jones is used as a template (see Figure 2 for an example). Face detection is executed independently of tracking and only transmits information on detected faces to tracking.

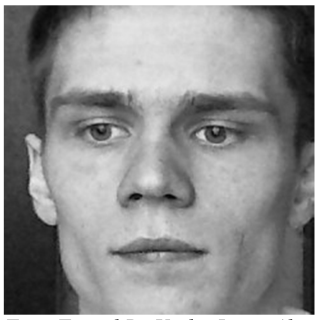

Figure 2: Face Found By Viola-Jones Algorithm And Used As A Template

4.2 Calculation of the Search Area

Looking for a new position of a face in the entire frame is impractical because, firstly, it can lead to errors, and secondly, it leads to additional calculations which negatively affects the performance. Therefore, the search area is chosen by to the following rule:

$$\begin{aligned} W &= 1.4w + 2|x_{i-1} - x_i| \\ H &= 1.4h + 2|y_{i-1} - y_i| \\ X &= x_i + (x_{i-1} - x_i) \\ Y &= y_i + (y_{i-1} - y_i) \end{aligned} \quad (5)$$

where

- x_i, y_i are the coordinates of the center of tracking area on frame i,
- w, h are the width and height of the temaplate respectively,
- X, Y are the coordinates of the center of the search area,
- W, H are the width and height of the search area, respectively.

Thus, the search area is 40% wider and taller than the template (

Figure 3). Additionally, the area is expanded in the direction of movement of the face (Figure 4). It proved to be enough for face tracking within natural human behavior.

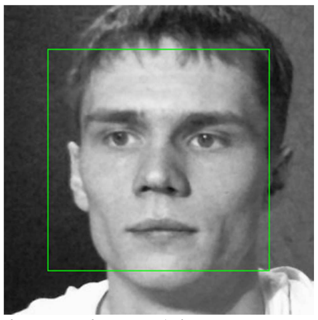
Figure 3: The Search Area (The Entire Image) And The Tracked Area (The Green Frame)

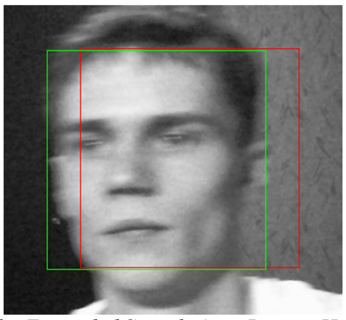
Figure 4: The Expanded Search Area During Horizontal Motion. The Red Frame Is The Previous Position, The Green Frame Is The Current One

4.3 Loss of Face

The best tracking performance can be achieved using either normalized cross-correlation or squared difference calculated using the Fourier transform. Since the normalized correlation is stable to global illumination changes, and its values lie within the [0; 1] interval, it was decided to use this method.

Tracking under certain conditions (changing facial expressions, face turn, etc.) can choose a wrong region (Figure 5), so it is necessary to choose a threshold value of R_{loss} (x, y) at which the face is considered lost.

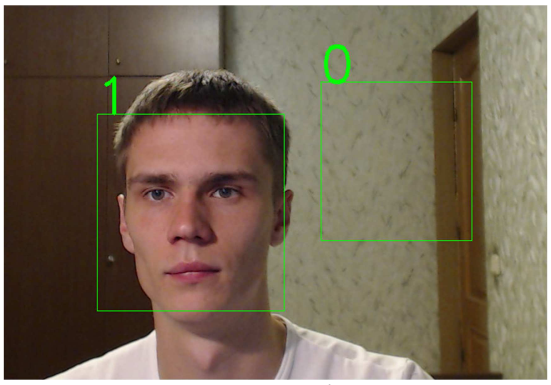

Figure 5: A Tracking Error

The loss threshold R_{loss} (x, y) was chosen empirically. The value was varied with a step of 0.01 to find a balance between preventing unnessessary losses on one hand and tracking after an error has occurred (and the area has slipped to the background) on the other hand. The optimal value was found to be R_{loss} (x, y) = 0.91.

4.4 Problems of the Basic Algorithm

The algorithm reacts poorly to any face changes with respect to the template, and in particular:

- changes in lighting;
- head turns;
- facial expressions;
- overlapping of faces with other objects;
- face scaling (a person can get closer to the camera or farther away).

That is, the basic template tracking does not operate stably enough, and modifications are needed in order to cope with the problems. These modifications are discussed in the next section.

5. TEMPLATE TRACKING MODIFICATIONS

5.1 Template Averaging

Template updates need to be done in order to overcome the problems associated with changes in lighting, head turns and any other changes to the face appearance. Two methods of updating the template are proposed.

1. The template on each frame is an average of previous N frames.

$$T_{i+1} = \frac{\sum_{j=1}^{N} T_{i-N+j}}{N} \tag{6}$$

2. The template is updated as a sum of the current template and the face identified in the current frame with certain coefficients. Thus, at each iteration the template is updated according to the following rule:

$$T_{i+1} = \alpha T_i + \beta F_i \tag{7}$$

where

- T_i is the template in the i-th frame;
- F_i is the image found by tracking on the i-th frame;
- α, β are the averaging coefficients, with $\alpha + \beta = 1$.

The second method showed better results as it is more responsive to changes. The values $\alpha = 0.8$ and $\beta = 0.2$ were experimentally chosen. These values provide a sufficiently rapid update of the template, and at the same time a tracking error in one frame does not lead to the background replacing the face in the template. Figure 6 provides some templates generated by this algorithm.

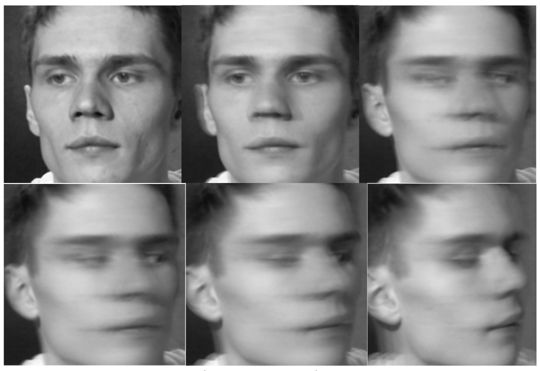
Figure 6: Updating Template Via An Averaging Filter In Case Of Head Rotation

Despite the fact that this modification copes with the problem of changes in facial expression, head rotation, etc., it also adds several issues:

- If a person comes out of the frame slowly, the tracking stops on the background near the border of the image.
- Tracking becomes prone to drifting in case of multiple head turns.
- In case of slow occlution of a face with large objects, the face is not considered lost.

5.2 Face Loss at the Edges of the Frame

To solve the problem with slow escape from the frame, the following solution is proposed:

- If an edge of the search area coincides with an edge of the frame, use only the opposite half of the template.
- If half of the face out of the frame, consider it lost.

This modification allowed to completely solve the problem with a slow escape out of the frame.

5.3 Scaling of Face by the Distance between the Eyes

To solve the problem with scaling template it has been proposed to use the distance between the eyes [11]. At the initial frame received from face detection one records templates of the eyes and the ratio of the distance between the eyes and the size of the face. Thereafter, at each new frame the eyes are found as well as the face. The whole face is scaled based on the change of the distance between the eyes. Furthermore, face angle is calculated depending on the eyes positions, and the search area is rotated by this angle (Figure 7).

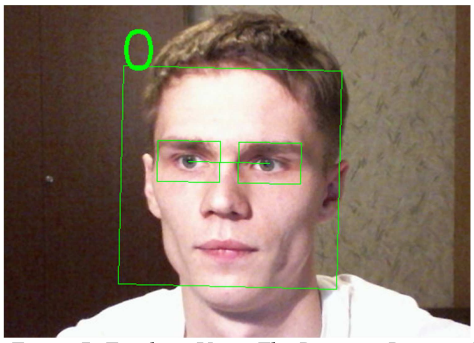

Figure 7: Tracking Using The Distance Between The Eyes

The main disadvantage of this modification is that an error in location of one of the eyes leads to wrong scaling and incorrect calculation of the angle of inclination, which leads to additional losses of tracking and affects the overall quality.

5.4 Scaling By Face Detection

Since the approach of using the distance between eyes is not stable enough, an alternative solution was suggested. When a person approaches the camera, they are usually facing the camera. Thus, the algorithm of Viola-Jones will detect faces of people coming up to the camera and stepping back from it, and it will allow us to update the template in time.

A face found by the Viola-Jones algorithm is considered an already known one and is used to update the template if the following three conditions are met:

- the area of intersection with the tracking area is more than 40% of the area of the smaller rectangle;
- the difference in size along one dimension is more than 10%. The algorithm of Viola-Jones always detects areas of the same shape, particularly a square in our case, so a 10% change in the length of a square's leg is equivalent to a 21% change of the area;
- the difference in size in each dimension is less than 20%, which is equivalent to 44% of the area.

Figure 8 shows a pair of rectangles that were considered matching.

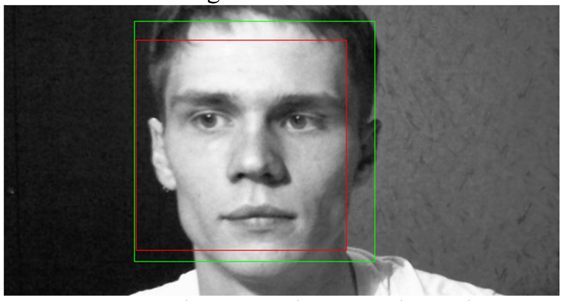
Figure 8: Updating Template Size. The Rred Frame Is The Previous Pposition Of The Tracking Area, The Green Frame Is A Face Detected By The Algorithm Of Viola-Jones

This solution also has a positive side effect: if the tracking region has slightly drifted off the face, a refresh of the template fixes this error.

5.5 Check By A Brightness Histogram

Because the tracking with template update may remain drift off to the background, some kind of check should be introduced to confirm the similarity of the current area with the person originally found.

A comparison of the current brightness histogram (Figure 9) with a reference one [42] can serve as such a check. Head rotation changes the contours and shape of the face dramatically, so it makes sense to use only the distribution of brightness in the tracking region.

A reference histogram is calculated when a new face is added to tracking.

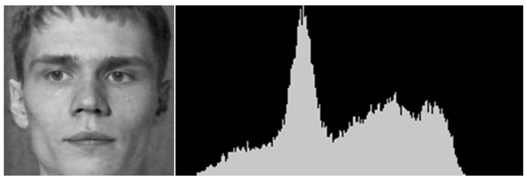
Figure 9: Template (Left) And The Brightness Histogram (Right)

Histograms can be compared using different methods [36]:

- normalized correlation

$$d(H_1, H_2) = \frac{\sum_I (H_1(I) - \bar{H}_1)(H_2(I) - \bar{H}_2)}{\sqrt{\sum_I (H_1(I) - \bar{H}_1)^2 \sum_I (H_2(I) - \bar{H}_2)^2}} \quad (8)$$

where

– $\bar{H}_k = \frac{1}{N}\sum_J H_k(J);$

– N is number of partition intervals.

- chi-square

$$d(H_1, H_2) = \frac{\sum_I (H_1(I) - H_2(I))^2}{H_1(I)} \quad (9)$$

– intersection

$$d(H_1, H_2) = \sum_I \min(H_1(I), H_2(I)) \quad (10)$$

– hellinger distance

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2}} \sum_I \sqrt{H_1(I) \cdot H_2(I)}} \quad (11)$$

The Hellinger distance was chosen as a measure of similarity because results of this measure lie in the interval [0; 1] and provide sufficient accuracy [47]. The Hellinger distance equals 0 for identical histograms and 1 for non-overlapping ones.

5.6 Checking The Histogram Color

As shown in [34], color information can be used to track a face quite accurately, so it can also be used instead of brightness histograms to verify the tracking region contents, as it better describes the object. The image is represented in the HSV format for this check, and the hue and saturation channels are used to construct a histogram.

The 2D histogram is partitioned into 18 bins along each channel. Such a reduced number of bins enables effective application of this method to noisy images, as well as increases the speed of histograms comparison.

When the color brightness is low (the value channel), the hue value is undefined, so the histogram is constructed with a restriction: pixels with brightness values lower than 10% of the total range are not taken into account. Further, hue is also undefined in case of low saturation, so pixels with saturation lower than 10% are also not counted.

The tracking area often includes background sections, resulting in worse description of the object of tracking. In order to minimize this effect the histogram is calculated for a subarea defined as follows:

$$\begin{aligned} x_h &= x + 0.15w \\ y_h &= y + 0.2h \\ w_h &= 0.7w \\ h_h &= 0.7h \end{aligned} \qquad (12)$$

where x, y are the coordinates of the top-left corner of the tracking rectangle; w, h are the width and height; x_h, y_h are the coordinates of the target rectangle's top-left corner; w_h, h_h are its width and height. Sample results are shown in Figure 10.

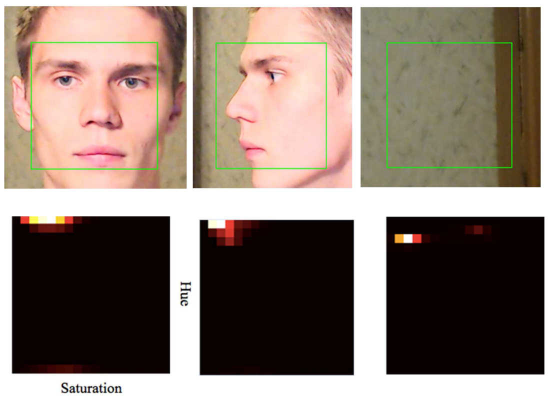

Figure 10: Color Histograms For Two Face Images And A Background Sample, Squares Denote The Subareas Used For Histogram Calculation. The Hellinger Distance From The First Histogram To The Second One Is $d(H_1,H_2)=0.42$, From The First To The Third One It's $d(H_1,H_3)=0.98$

The loss threshold was experimentally defined as $d_{loss}(H_1, H_2) = 0.7$.

5.7 Recovery of a Lost Face

Often when a face is lost by mistake, it is soon re-found by the face-detection algorithm. This can be used to restore a lost face with the same ID if a newly detected face is accepted to be the same one.

A face is considered an already known one under the following conditions:

- Less than a second passed since the loss.
- The distance between the lost and the detected faces meets the following condition:

$$d(F_{lost}, F_{new}) < \frac{w_{lost} + h_{lost} + w_{new} + h_{new}}{4} \qquad (13)$$

where

- – F_{lost} is the rectangle of the lost face with width w_{lost} and height h_{lost},
- – F_{new} is the rectangle of the new face with width w_{new} and height h_{new}.

- The size of the face did not change dramatically:

$$\left|1 - \frac{S(F_1)}{S(F_2)}\right| < 0.4 \qquad (14)$$

where S(F) is the rectangle area.

- The face was not lost near a frame edge.

5.8 Scaling of Large Areas

Since matching a large template to a large area is quite a heavy operation and the detailization can be excessive, the template and the search area can be scaled down when they are larger than a certain useful size. Experimentally the values were chosen as follows: the template is reduced to 150x150 pixels if it is originally larger than 170x170 pixels; the search area is scaled accordingly. The scaling is done by bilinear interpolation as it yields acceptable accuracy and a sufficiently high-speed computation.

6. EVALUATION OF THE QUALITY OF THE SYSTEM

Currently, we are not aware of any universal methods to evaluate the quality of the developed system for collecting statistics on visitors, therefore a new software tool was developed that allows to gauge the system's performance.

The main idea of this software tool is the use of a test video with reference markings. Each video is pre-marked manually by an expert. Thereafter, the reference marking is compared with the results of the system and certain quality indicators are calculated. Thus, the software consists of two parts: video marking and comparison of an actual system's performance with a reference.

6.1 Video Marking

In this stage, we are given a video record, and we have to mark each face in every frame with a bounding rectangle. A similar problem is addressed by a Caltech labeling tool [48], however this tool does not quite fit the case of highly dynamic objects in front of a static camera. It is mainly made for annotating videos taken from a moving car in which case the camera motion dominates objects' own motion. For this reason, Dollar et al. made the choice to assist labeling by simply interpolating the rectangles between manually labeled key frames. In our case, on the other hand, objects' motion is highly nonlinear, which whould require a large number of key frames to describe. Thus, we chose to employ an assisting tracking mechanism: once the operator has pointed out a face in one frame, the program can track the rectangle contents in subsequent (or preceeding) frames with the operator checking its results and adjusting the location and size of the rectangle. The user interface implemented for this task is shown in

Figure 11.

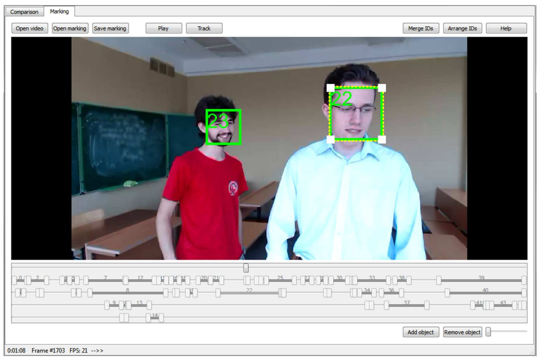

Figure 11: The Video Marking User Interface

However, marking a video for assessment of a system like ours still gets problematic due to several ambiguities:

- There are times when it is not clear at which point a face can be considered gone, for example, when it partially exits the frame, or when the person is turning away from the camera. If we mark the face visible as long as we can see at least a part of it that would lead to overestimating the tracking error: it may not be required to track faces that are hardly visible. If, on the other hand, we mark the face disappearance too early, it will not be possible to tell if the remaining frames were the tracking doing better than we expected or its failure to detect the face disappearance.
- The same applies to faces coming into view: it may not be required to detect faces that are not completely visible but if a face stays in such an indetermined state during all its "lifetime" and we do not mark its appearance at all, a detection that may still occur will be indistinguishable from a false detection.
- Another complication with face detection is the distance at which the system is designed to operate: if a face stays near the target distance line it is also not easy to decide whether it should be "visible".

A perfect way to overcome these ambiguities would be including a lot of additional information in the reference marking of a video: the distance to each face, its orientation, its occlusion state, etc. This way it would be possible to find rotation angles at which a face gets detected/lost, etc., and to make accurate assessment of the tracking quality. However, this way requires much more marking work, which does not seem realistic at the moment.

Therefore, we employed a more rough solution. First, we tune our tracking system to detect as many faces as possible and to track them as long as possible, thus allowing false positives and "over-tracking" when a face has gone. Then the system's output for a video is taken as an initial estimate and is corrected by an operator (removing false positives and adding what was clearly missed) to produce a reference marking.

6.2 Comparison With The Reference

As said before, reference markings and the system's outputs consist of face positions recorded at each frame of a video. The first step in assessing how close the system's results are to the reference is to find matching faces in the two markings. Each face in both sets is assigned a unique identifier (ID), and then the following procedure is used to establish the relations between them:

- A 'one-to-many' matching is established between the identifiers (one face from the reference can correspond to several faces from the system's results because of the face recovery mechanism).
- At frame f face r from the reference is considered to match face t from the tracking results if

$$Inters(f, r, t) > c \qquad (16)$$

Journal of Theoretical and Applied Information Technology
31st January 2015. Vol.71 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$Inters(f,r,t) = \frac{S(R_r^f \cap R_t^f)}{\min\left(S(R_r^f), S(R_t^f)\right)} \quad (17)$$

where

- R_r^f is the rectangle of the reference face;
- R_t^f is the rectangle of the face from tracking;
- $S(R)$ denotes the area of a rectangle R;
- c is a threshold value. We used the value $c = 0.3$.

- A pair of identifiers are considered to match if the corresponding faces match (as defined by (16)) at least in one frame. If a face in the tracking results intersects multiple faces from the reference, the best match is determined by maximizing the following function:

$$\max_r \sum_f Inters(f,r,t) \quad (18)$$

6.3 Quality Indicators

There may be many ways to qualitatively express the quality of tracking and the system as a whole.

- First, the most practically important result is how accurately the total number of faces is counted. We express the positive or negative difference as a percentage of the real (reference) number of faces. This indicator is a combination of several error types that can partially compensate each other but the number of interest is the total number of faces. The error types are
  - faces that were not detected;
  - false positive detections;
  - faces counted multiple times under different IDs;
  - faces that were detected but erroneously assigned an ID of a previously lost face.
- Another practical result is the average time a person spends in front of the camera. Therefore, the second indicator is a mean time in frame error expressed as a percentage of the reference mean time. Again, multiple error types contribute to this indicator:
  - face detection delay;
  - tracking loosing faces too early;
  - tracking failing to notice a face disappearance instantly.

The later two error types are affected directly by the proposed modifications; therefore, we consider them as separate indicators.

- Mean under-tracking time is calculated as follows:

$$\left(\sum_{\{i:\, t_{end_i}^r > t_{end_i}^t\}} t_{end_i}^r - t_{end_i}^t \right) \Big/ N \quad (19)$$

where

- $t_{end_i}^r$ is the i'th face's disappearance time in the reference marking;
- $t_{end_i}^t$ is the i'th face's disappearance time in the tracking results;
- N is the total number of faces that were matched between the two markings.

- Mean over-tracking time is calculated as follows:

$$\left(\sum_{\{i:\, t_{end_i}^r < t_{end_i}^t\}} t_{end_i}^t - t_{end_i}^r \right) \Big/ N \quad (20)$$

where the symbols are defined as above.

7. EXPERIMENTAL RESULTS

Experiments were conducted on 3 videos about 2 minutes long. The videos are quite challenging: several people are walking around, both in the foreground and the background, they cover each other, occlude their faces with their hands, and so on (like in Figure 12). In addition, some compression artifacts are present in the videos. Table 1 shows some parameters of test videos.

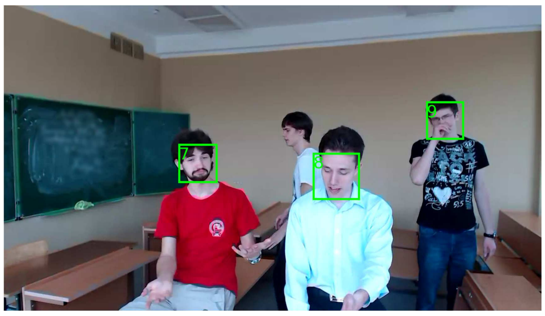
Figure 12: A Frame From Test Video 3

Table 1: Parameters Of The Test Videos

Video #	Duration	Number of faces in reference	Average time of face's presence in the frame, sec
1	1:30	41	3.34
2	2:30	76	5.51
3	2:29	48	5.12
Total	6:29	165	4.86

In the experiments, we measure the quality indicators defined in the previous section for different configurations of the tracking algorithm. We test the basic template matching algorithm without our modifications, then we try enabling every modification separately, then we test the configuration with all modifications enabled and then try disabling every one of them. The results for all 3 videos together are shown in Table 2.

As expected, we can see that enabling all modifications improves the results of the basic algorithm. We can also note that every modification at least doesn't worsen the results. Particularly, scaling down large templates doesn't significantly improve the indicators (and even adds a couple percents to mean time in frame error, which can be considered "noise" in the measurements) because it doesn't directly affect the algorithm. However, in case the computer is low-performance enough or the number of people in the frame is large enough, scaling can also affect the quality indicators by allowing the tracking to maintain a higher framerate and thus to skip less frames.

Tables 3–

Table 5 present the results for individual videos.

Table 2: The Effects Of Tracking Modifications. Overall Results On 3 Test Videos

Template averaging	Cropping on edges	Updating on redetection	Histogram check	Restoration	Scaling	Total face count error (%)	Mean time in frame error (%)	Mean under-tracking time, ms	Mean over-tracking time, ms
✓	✓	✓	✓	✓	✓	+1	-11	223	359
—	✓	✓	✓	✓	✓	+16	-30	420	278
✓	—	✓	✓	✓	✓	-2	-12	276	344
✓	✓	—	✓	✓	✓	0	-9	295	385
✓	✓	✓	—	✓	✓	+15	-18	543	999
✓	✓	✓	✓	—	✓	+29	-30	247	371
✓	✓	✓	✓	✓	—	-1	-9	269	359
✓	—	—	—	—	—	+139	-59	566	764
—	✓	—	—	—	—	+315	-72	682	1419
—	—	✓	—	—	—	+272	-81	724	175
—	—	—	✓	—	—	+53	-45	459	82
—	—	—	—	✓	—	+59	-54	705	236
—	—	—	—	—	✓	+318	-82	705	211
—	—	—	—	—	—	+199	-71	628	577

The first video contains the most rapidly alternating faces, average time of presence in the frame is 3.34 sec. On the other hand, the number of faces simultaneously present in the frame is smaller than in the second and third videos. Also, the video has inhomogeneous background.

The second video also has an inhomogeneous background; people stay in the frame longer but they are often very close to each other.

The third video has a more homogeneous background, and the number of people simultaneously in the frame is much smaller than in the second video.

Table 3: The Effects Of Tracking Modifications On Video 1

Template averaging	Cropping on edges	Updating on redetection	Histogram check	Restoration	Scaling	Total face count error (%)	Mean time in frame error (%)	Mean under-tracking time, ms	Mean over-tracking time, ms
✓	✓	✓	✓	✓	✓	-2	-4	120	113
—	✓	✓	✓	✓	✓	+5	-19	185	142
✓	—	✓	✓	✓	✓	-7	-2	146	99
✓	✓	—	✓	✓	✓	+5	-10	124	104
✓	✓	✓	—	✓	✓	0	-10	442	23
✓	✓	✓	✓	—	✓	+32	-29	119	114
✓	✓	✓	✓	✓	—	-5	-1	103	114
✓	—	—	—	—	—	+193	-69	476	75
—	✓	—	—	—	—	+290	-71	509	1146
—	—	✓	—	—	—	+285	-78	659	328

—	—	—	✓	—	—	+54	-44	191	77
—	—	—	—	✓	—	+22	-28	549	327
—	—	—	—	—	✓	+295	-78	553	327
—	—	—	—	—	—	+141	-55	463	948

Table 4: The Effects Of Tracking Modifications On Video 2

Template averaging	Cropping on edges	Updating on redetection	Histogram check	Restoration	Scaling	Total face count error (%)	Mean time in frame error (%)	Mean under-tracking time, ms	Mean over-tracking time, ms
✓	✓	✓	✓	✓	✓	+1	-11	341	143
—	✓	✓	✓	✓	✓	+24	-33	544	83
✓	—	✓	✓	✓	✓	-1	-11	352	139
✓	✓	—	✓	✓	✓	0	-6	315	286
✓	✓	✓	—	✓	✓	+25	-26	734	1346
✓	✓	✓	✓	—	✓	+37	-34	371	169
✓	✓	✓	✓	✓	—	-1	-8	367	170
✓	—	—	—	—	—	+162	-68	749	668
—	✓	—	—	—	—	+429	-83	922	1901
—	—	✓	—	—	—	+357	-87	906	0
—	—	—	✓	—	—	+62	-45	519	56
—	—	—	—	✓	—	+79	-67	943	1
—	—	—	—	—	✓	+437	-89	928	0
—	—	—	—	—	—	+292	-82	837	260

Table 5: The Effects Of Tracking Modifications On Video 3.

Template averaging	Cropping on edges	Updating on redetection	Histogram check	Restoration	Scaling	Total face count error (%)	Mean time in frame error (%)	Mean under-tracking time, ms	Mean over-tracking time, ms
✓	✓	✓	✓	✓	✓	+2	-15	135	76
—	✓	✓	✓	✓	✓	+13	-33	435	75
✓	—	✓	✓	✓	✓	+2	-18	245	71
✓	✓	—	✓	✓	✓	-4	-12	175	73
✓	✓	✓	—	✓	✓	+13	-9	325	1305
✓	✓	✓	✓	—	✓	+15	-25	170	75
✓	✓	✓	✓	✓	—	+2	-15	135	75
✓	—	—	—	—	—	+58	-19	345	1531
—	✓	—	—	—	—	+154	-38	464	917
—	—	✓	—	—	—	+127	-65	505	318
—	—	—	✓	—	—	+40	-46	520	34
—	—	—	—	✓	—	+60	-43	479	534
—	—	—	—	—	✓	+150	-64	498	444
—	—	—	—	—	—	+100	-48	441	718

8. CONCLUSION

Template-matching tracking was used as a core method in this work. The following modifications were proposed to improve its performance:

- Template averaging to allow the tracking to adapt after head rotation and other changes in the object.
- Using a cropped template at frame boundaries to handle faces that partially leave the frame.
- Scaling a face using the results of face detection to handle distance changes.
- Check by color histograms to detect faces disappearance with more confidence.
- Recovery of lost faces to prevent counting a single person multiple times.
- Downscaling large templates to improve speed for faces close to the camera.

A special software tool was developed to test the algorithm's performance. It is based on marking a video by hand and comparing the system's results to the reference. Experiments were conducted for the algorithm with all modifications, without any and with some modifications enabled. The results proved the effectiveness of the modifications.

Directions of future work include both development of better tracking algorithms and improving the quality assessment method: more detailed video marking may allow more meaningful quality indicators to be calculated.

REFERENCES:

[1] Ramer, J., Soroca, A., Doughty, D. 2013, Rendering targeted advertisement on mobile communication facilities. United States Patent 8364521. Pub. date 01/29/2013. URL: http://www.freepatentsonline.com/8364521.html.

[2] Van Datta, G., Zalewski, G., 2007, Targeted advertising. WIPO Patent Application WO/2007/041022. Pub. date 04/12/2007. URL: http://www.freepatentsonline.com/WO2007041022A2.html.

[3] Tang, K. 2012, Advertisement delivery system with location based controlled priority mechanism and method of operation thereof. United States Patent Application 20120150853. Pub. date 06/14/2012. URL: http://www.freepatentsonline.com/y2012/0150853.html.

[4] Tung, K. W., Chan, R. E. 2011, Advertising system and method. United States Patent Application 20110071911. Pub. Date 03/24/2011. URL: http://www.freepatentsonline.com/y2011/0071911.html.

[5] Miyagi, Y. 2000, Video appearing mirror. Japanese Patent JP2000112418. Pub. Date 04/21/2000. URL: http://www.freepatentsonline.com/JP2000112418.html.

[6] Types of counting systems [electronic resource] / / Counters [site]. URL: http://www.uchet.biz/vidy-sistem-podscheta-posetitelei.php (date of access: 12.06.2013).

[7] Ernst A.. Ruf T. Kublbeck C. A modular framework to detect and analyze faces for audience measurement systems // Proc. GI Jahrestagung, 2009. pp.3941-3953.

[8] Audience Engagement [Электронный ресурс] // Intel AIM Suite [сайт]. URL: https://aimsuite.intel.com/aim-suite-work/audience-engagement (дата обращения: 12.06.2013).

[9] Viola P. Jones M.J. Rapid Object Detection using a Boosted Cascade of Simple Features // IEEE Conf. on Computer Vision and Template Recognition (CVPR 2001), 2001.

[10] Viola P. Jones M.J. Robust real-time face detection // International Journal of Computer Vision, vol. 57, no. 2, 2004. pp.137-154

[11] Dyakov V.P. Wavelets. From theory to practice. Moscow.: SOLON-Press, 2004. 440 p.

[12] Šochman, Jan; Matas, Jiří (2004). Adaboost with Totally Corrective Updates for Fast Face Detection.

[13] Lienhart R. Maydt J. An Extended Set of Haar-like Features for Rapid Object Detection // IEEE ICIP 2002, Vol. 1, pp. 900-903, 2002

[14] H. Jin, Q. Liu, H. Lu, and X. Tong. Face detection using improved LBP under Bayesian framework. In Proc. Third International Conference on Image and Graphics (ICIG), pages 306-309, Hong Kong, China, 2004

[15] Rodriguez Y. Face Detection and Verification using Local Binary Templates: PhD Thesis / École Polytechnique Fédérale de Lausanne, 2006. 150 p.

[16] Chang-yeon J. Face detection using lbp features // Stanford University, Final Project Report, 2008

[17] Kryvtsov O.A. Methods, algorithms and software system for tracking heads of people in video frames based on geometric texture models: thesis for the degree of doctor in technical sciences / Tomsk State University of Control Systems and Radio Electronics (TSUCSRE); National Research Tomsk Polytechnic University. Tomsk, 2010. 202 p.

[18] Casas D. Real time face tracking methods: Master's Thesis. / Universitat Autònoma de Barcelona, 2009. 90 p.

[19] Roy-Chowdhury R., Xu Y. Face Tracking // Encyclopedia of Biometrics, Springer. 2009. P. 383-388.

[20] Mahalakshmi T., Muthaiah R., Swaminathan P. Review Article: An Overview of Template Matching Technique in Image Processing // Research Journal of Applied Sciences, Engineering and Technology, 2012. Vol. 4 (24). P. 5469-5473.

[21] Friemel B., Bohs L., Trahey G. Relative performance of two-dimensional speckle-tracking techniques: Normalized correlation, non-normalized correlation and sum absolute difference // IEEE Ultrasonics Symposium. 1995. Vol. 2. P.1481-1484.

[22] Lewis J.P. Fast template matching // Vision Interface. 1995. Vol. 95. P. 120-123.
[23] Barnea D.I., Silverman H.F. A class of algorithms for fast digital image registration // IEEE Trans. Comput. Vol. C-21, 1972. P. 179-186.
[24] Brunelli R. Template Matching Techniques in Computer Vision: Theory and Practice. – Wiley, 2009. 348 pages.
[25] Martin J., Crowley J.L. Experimental Comparison of Correlation Techniques // International Conference on Intelligent Autonomous Systems, 1995.
[26] Ifeachor E., Jervis B. Digital Signal Processing: Practical Approach, 2nd Edition / translated from English by Doroshenko I.Y, Nazarenko A.V., Moscow: Williams, 2004. 992 pages.
[27] Lucas B., Kanade T. An iterative image registration technique with an application to stereo vision. // In Proceedings of the International Joint Conference on Artificial Intelligence. 1981. P. 674–679.
[28] Matthews I., Ishikawa T., Baker S. The template update problem // IEEE Transactions on Template Analysis and Machine Intelligence, 2004. Vol. 26. No. 6. P. 810-815.
[29] Nguyen H.T., Worring M., van den Boomgaard R. Occlusion robust adaptive template tracking // Proc. IEEE Conf. Computer Vision, ICCV, 2001. Vol. 1. P. 678 -683.
[30] Baker S., Matthews I. Lucas-Kanade 20 Years On: A Unifying Framework // International Journal of Computer Vision, 2004. Vol. 56. No. 3. P. 221 - 255.
[31] Tomasi C., Kanade T. Detection and Tracking of Point Features / Carnegie Mellon University Technical Report CMU-CS-91-132, 1991.
[32] Shi J., Tomasi C. Good features to track // IEEE Conference on Computer Vision and Template Recognition, 1994. P. 593-600.
[33] Bouguet J.-Y. Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm / Intel Corporation Microprocessor Research Labs, 2000.
[34] Bradski G.R. Computer vision face tracking as a component of a perceptual user interface // Intel Technology Journal Q2 1998, 1998.
[35] Boyle M. The effects of capture conditions on the CAMSHIFT face tracker // Technical Report, Department of Computer Science, University of Calgary, Alberta, Canada, 2001.
[36] Exner D., et al. Fast and Reliable CAMShift Tracking: PhD Thesis / Bauhaus University, Wiemar, 2009.
[37] Guraya F.F.E., Bayle P.-Y., Cheikh F.A. People Tracking via a Modified CAMSHIFT Algorithm, 200
[38] Allen J.G. et al. Object tracking using CAMSHIFT algorithm and multiple quantized feature spaces // Pan-Sydney area workshop on Visual information processing, 2004. P. 3-7.
[39] Cascia M. L. Sclaroff S. Athitsos V. Fast, Reliable Head Tracking under Varying Illumination: An Approach Based on Registration of Texture-Mapped 3D Models // IEEE Trans. Template Anal. Mach. Intell., 2000. Vol. 22 (4). P. 322-336.
[40] Ahlberg J. *CANDIDE-3 -- an updated parameterized face.* Report No. LiTH-ISY-R-2326, Dept. of Electrical Engineering, Linköping University, Sweden, 2001.
[41] Martins P. Parametric Face Alignment: Generative and Discriminative Approaches: PhD Thesis / University of Coimbra, 2012.
[42] Vrazhnov D.A., Shapovalov A.V., Nikolaev V.V. On the quality of object tracking algorithms in video. Computer Research and Modeling, 2012. T. 4. № 2 pp. 303-313.
[43] Bradski G., Kaehler A.. Learning OpenCV. Sebastopol: O'Reilly Media. P. 194-221.
[44] OpenCV computer vision library. OpenCV, 2013. URL: http://opencv.org/.
[45] JsonCpp library. JsonCpp - JSON data format manipulation library, 2013. URL: http://jsoncpp.sourceforge.net/.
[46] Myers G., Badzhett T. Sandler K. Art of program testing. Translated from English by A.Guzkevich. 3rd ed. Moscow, Dialectics, 2012. 272 p.
[47] Kaner K., Falk D. Software Testing. Fundamental concepts of business applications management. Kiev: Diasoft, 2001. 544 p.
[48] Dollar, Piotr and Wojek, Christian and Schiele, Bernt and Perona, Pietro. Pedestrian detection: An evaluation of the state of the art. IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 34, number 4, 2012. Pages 743-761.