Overlaying signs of drug use on images of users faces in real time

Uladzimir Kavalenka, Dmitriy Kostiuk, Stanislav Derechennik-jr.

Brest State Technical University, 267, Moskovskaya Str., vl.kovalenko1989@yandex.ru

Abstract: *The software is presented for applying the signs of prolonged use of illicit drugs of various groups and classes to a photograph of a user's face in order to prevent their use. Used image processing algorithms are discussed upon with the obtained quality results and resource consumption.*

Keywords: Face detection, transformation, morphing. Delaunay triangulation, alpha blending.

1. INTRODUCTION

With the development of facial recognition technology, a new class of tasks related to the modification of a user's face photo has emerged. Such tasks arise mainly in the entertainment and media industry, as well as for the purposes of propaganda and public awareness. Basically, it is the imposition of various features typical of certain age or social categories on the face.

Such examples are Internet services on "aging" of a user's face (http://cherry.dcs.aber.ac.uk/Transformer/), combining the face of a person and the characters of movies and games, predicting the appearance of the future child based on the faces of his parents and the substitution of faces with preservation of tonality and color (https://photofunia.com/ru/effects/face_swap). The most interesting are the projects on neural network use to insert the desired face into the video while preserving facial expressions (DeepFake).

In the project described in the present work, the task of applying the signs of prolonged use of illicit drugs of various groups and classes to a photograph of a user's face was solved to prevent their use. It should be executed on a time scale close to real (with minimal delays). The source of the image is the webcam of the all-in-one computer running the software. The closest analogue can be considered a license for alcohol, issued in some Arab countries, where the face in the photo is changed - the consequences of excessive alcohol consumption are applied to it.

2. INPUT DATA

The software had to work on a monoblock computer with a touch screen, with a 7th generation Intel Core i5 central processor, 4 GB of RAM and an integrated Intel UHD 620 graphics accelerator running Windows 10 Home. The next important feature is the impossibility of leasing or acquiring additional server capacities (local or cloud), i.e. processing should be done only on the local computer, which imposes significant restrictions on the possible methods of solving the problem.

The analysis of the task allowed us to distinguish the following stages: face detection in a video stream; capturing a static frame with the face in the most appropriate pose; application of characteristic features to the corresponding parts of the face (ulcers, wrinkles, bruises, ruts, unnatural color, tumors).

Currently, the dominant tool for tasks of this class is the use of artificial intelligence (neural networks). A pre-trained neural network would have done a really good job with this task, but there were several significant and unrecoverable problems: extremely limited development time, not enough for full network training; lack of sufficient data for training (photos of people before and after a significant period of use of prohibited substances); insignificant hardware power (for training and for real work); inability to use a remote server to process captured images.

Thus, the method of combining faces became the only possible option.

3. GENERAL METHOD DESCRIPTION

The algorithm of the system is as follows: the presence of a human face is detected in the video stream (there should be only one face in the frame), then the most suitable head position is waited for (for the best result of the overlay algorithm). Further, one of the masks is selected at random (a somewhat modified photograph of a person who has used drugs for a long time with a well-visible close-up face) and an oval of the face is highlighted. The cut face is transformed to fully fit the face of a person in shape, size and angle. The last step is to insert the transformed face into the original photo with pixel mixing and color correction (optional, but required for the naturalness of the resulting image and the absence of artifacts on the face-background border).

Photos distributed under a free license by one of the US Drug Enforcement Administrations of Texas, USA, were used as masks.

4. TOOLS AND LIBRARIES

The main development language of the project was C# and the .NET 4.7 platform (the use of software is assumed exclusively on Windows). Open source libraries were used for image processing, recognition and operations with graphic objects. DlibDotNet - C# wrapper for the Dlib library (machine learning, face detection in the image); EmguCV - OpenCV implementation in C#, components for displaying images and a video stream; OpenCvSharp is a C# shell above the OpenCV library.

With the exception of Windows Forms, used as a graphical framework and for input processing, all components are distributed under free licenses and are available on GitHub.

5. FACE DETECTION IN A VIDEO STREAM

Each video frame is first displayed in the corresponding window element, and at the same time it

is processed by a method that analyzes it for the presence of a person's face. For this, the Viola-Jones method is used, which is fundamental for the real-time search of objects in the image [1].

This method searches for faces and facial features according to the general principle of a sliding window. The image is scanned by the search window and then the classifier is applied to each position. The system of training and selection of the most significant features is fully automated and does not require human intervention, so this approach works quickly. The training of the classifier is very slow, but the results of the face search are very fast, which is why this method of face recognition was chosen. Viola-Jones method is one of the best in terms of performance indicators of recognition / speed of work. Also, this detector has an extremely low probability of false detection of the face. The algorithm works well and recognizes facial features at a slight angle, up to about 30α. At an inclination angle of more than 30°, the percentage of detections drops sharply [2].

In order to perform any actions with data, an integral representation of images [3] is used in the Viola-Jones method. Such a representation is often used in other methods, for example, in wavelet transforms, SURF, and many other algorithms. The integral representation allows one to quickly calculate the total brightness of an arbitrary rectangle on a given image, and no matter what the rectangle is, the calculation time is constant.

The integral representation of the image is a matrix that has the same size as the original image. Each element of it contains the sum of the intensities of all pixels that are to the left and above the element. The elements of the matrix are calculated according to the following formula:

$$L(x,y) = \sum_{i=0,j=0}^{i\le x,j\le y} I(x,y) \qquad (1)$$

where $I(i,j)$ is intensity of the pixel on the original image.

Each matrix element is the sum of pixels in a rectangle from (0,0) to (x,y), i.e. the value of each (x,y) pixel is equal to the sum of the values of all pixels to the left and above the given (x,y) pixel. The matrix calculation takes linear time proportional to the number of pixels in the image; therefore, the integral image is calculated in one pass.

Using such an integral matrix, one can very quickly calculate the sum of the pixels of an arbitrary rectangle, of arbitrary area.

In the context of the algorithm, there are many objects (images) that are somehow divided into classes. A finite set of images is given for which it is known which class they belong to (for example, this could be the "frontal position of the nose" class). This set is called a training set. Class affiliation of other objects is not known. It is required to construct an algorithm capable of classifying an arbitrary object from the original set [4].

In machine learning, the task of classification refers to the section of training with a teacher when classes are divided. Pattern recognition is essentially a classification of images and signals. In the case of the Viola-Jones algorithm for face identification and recognition, the classification is a two-class one.

The formulation of the classification problem is as follows:

There is X – a set in which the description of objects is stored, Y – a finite set of numbers belonging to classes. There is a dependency between them – a mapping $Y: X \to Y$. The training set is presented as $X_m = \{(x_1, y_1), \dots, (x_m, y_m)\}$. A function f of the feature vector X, which gives the answer for any possible observation X and is able to classify the $x \in X$ object. This simple rule should work well on new data too.

For detection, the EmguCV library and the DetectMultiScale method were used, the task of which is to detect the presence of a face in the image. It is important to note that the parameters of the method were set in such a way that the detection was carried out most carefully. This made it possible to quickly and efficiently implement a peculiar determination of the position of the head (if the head is turned relative to the camera at an excessively large angle, detection does not occur, and the next frame is processed). Also, further processing is not performed if several persons are detected in the frame to eliminate the need to sort the faces and determine the required face.

The next step is to transfer the frame for processing to the Dlib library to search for 68 control points (this was quite enough for the project's purposes, because they were able to detect the necessary part of the face with acceptable accuracy). A pre-formed "standard" data set "shape_predictor_68_face_landmarks" provided by library developers was used (to save time and because it was impossible to conduct full training).

The result of the work is an array with positions of 68 control points that rather accurately outlines the face of a person in the image. A schematic result of the work is presented in the figure 1.

6. DELAUNAY TRANSFORMATION

The task of constructing the Delaunay triangulation (an example of such triangulation is presented in the figure) is one of the basic tasks in computational geometry. Many other tasks are reduced to it; it is widely used in computer graphics and geo-information systems for surface modeling and solving spatial problems.

A convex triangulation is such a planar graph, all internal areas of which triangles are, and the minimal polygon covering all triangles is convex. Triangulation is called a Delaunay triangulation if it is convex and satisfies the Delaunay condition, i.e. if none of the given points of the triangulation fall inside the circle described around any constructed triangle.

All iterative algorithms for the Delaunay triangulation are based on a very simple idea of sequentially adding points to partially constructed Delaunay triangulation. Formally, it looks like this.

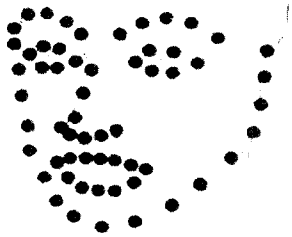

Fig.1 – Control points

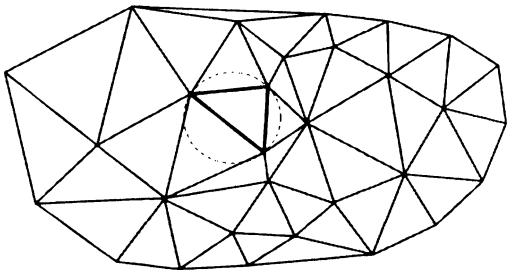

Fig.2 - Example of Delaunay triangulation

A set of N points given.

Step 1. On the first three reference points we build one triangle.

Step 2. In the n iterations loop for all other points, perform steps 3-5.

Step 3. The next n-th point is added to the already constructed structure of triangulation as follows. First, the point is localized, i.e. there is a triangle (built earlier) in which the next point falls. Or, if the point does not fall inside the triangulation, there is a triangle on the border of the triangulation, closest to the next point.

Step 4. If a point falls on a previously inserted triangulation node, then such a point is usually discarded, otherwise the point is inserted into triangulation as a new node. Moreover, if a point falls on a certain edge, the edge is split into two new ones, and both triangles adjacent to the edge are also divided into two smaller ones. If a point gets strictly inside some triangle, it is divided into three new ones. If a point falls outside the triangulation, then one or more triangles are constructed.

Step 5. Local checks of newly obtained triangles for compliance with the Delaunay condition are carried out and necessary rebuilds are performed.

One of the most important operations performed when triangulation is built is to check the Delaunay condition for given pairs of triangles. In practice, verification through the equation of the circumcircle is usually used.

The equation of a circle passing through $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ points can be written as equation (1).

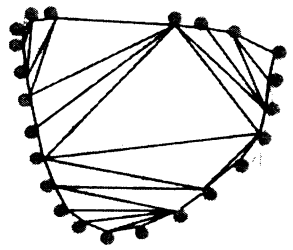

Fig.3 – Delaunay triangulation

$$\begin{vmatrix} x^2+y^2 & x & y & 1 \\ x_1^2+y_1^2 & x_1 & y_1 & 1 \\ x_2^2+y_2^2 & x_2 & y_2 & 1 \\ x_3^2+y_3^2 & x_3 & y_3 & 1 \end{vmatrix} = 0 \qquad (2)$$

or $(x^2+y^2)a - xb + yc - d = 0,$

where $a = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}, b = \begin{vmatrix} x_1^2+y_1^2 & y_1 & 1 \\ x_2^2+y_2^2 & y_2 & 1 \\ x_3^2+y_3^2 & y_3 & 1 \end{vmatrix},$

$c = \begin{vmatrix} x_1^2+y_1^2 & x_1 & 1 \\ x_2^2+y_2^2 & x_2 & 1 \\ x_3^2+y_3^2 & x_3 & 1 \end{vmatrix}, d = \begin{vmatrix} x_1^2+y_1^2 & x_1 & y_1 \\ x_2^2+y_2^2 & x_2 & y_2 \\ x_3^2+y_3^2 & x_3 & y_3 \end{vmatrix}.$

Then the Delaunay condition for any given $\Delta((x_1, y_1), (x_2, y_2), (x_3, y_3))$ triangle will be fulfilled only when for any (x_0, y_0) triangulation node there will be $(a(x_0^2+y_0^2) - bx_0 + + cy_0 - d)sgn\, a \geq 0$, i.e. when (x_0, y_0) is not inside the circle described around the $\Delta((x_1, y_1), (x_2, y_2), (x_3, y_3))$ triangle. To simplify the calculations, it can be noted that if the $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ triple of points is right (i.e., they are traversed in a triangle clockwise), then $sgn\, a = -1$, and vice versa, if the triple is left, then $sgn\, a = 1$.

The direct implementation of such a verification procedure requires 29 multiplication and squaring operations, as well as 24 addition and subtraction operations [6].

The complexity of this algorithm includes the difficulty of finding the triangle, to which a dot is added at the next step, the complexity of building new triangles, and the complexity of the corresponding rebuilding of the triangulation structure as a result of unsatisfactory checks of the neighboring triangles of the obtained triangulation for the fulfillment of the Delaunay condition.

The complexity of this task is $O(NlogN)$. There are algorithms that achieve this estimate in average and worst cases. In addition, algorithms are known that allow, in some cases, to achieve $O(N)$ on average.

Other options for checking the Delaunay condition are also possible, such as checking with a previously calculated circumference, checking the sum of opposite angles, modified check of the sum of opposite angles.

The purpose of the transformation is to divide the face image into triangles for transformation and subsequent imposition on the user's face. The transformation is necessary to fully match the size of the part of the image, the angle of rotation and position, which allows to achieve the best quality of the resulting image. A schematic result is shown in Figure 3.

7. AFFINE TRANSFORMATIONS

Affine transformation is a mapping of a plane or space into itself, in which parallel straight lines go into parallel straight lines, intersecting patterns — into intersecting straight lines, crossed straight lines — into crossed straight lines [7].

Affine transformation $f: \mathbb{R}^n \to \mathbb{R}^n$ is a transformation of the $f(x) = Mx + v$, where v is a reversible matrix (non-singular affinor) and $v \in \mathbb{R}^n$.

Affine transformation can be obtained as follows:

1. Select a new space basis with a new origin v.

2. Each point x in space should be associated with a point $f(x)$, that has the same coordinates relative to the new coordinate system as in the old one.

Examples of affine transformations are motion, tension / compression, and similarity transformation.

After Delaunay triangulation, an affine transformation is performed for each triangle using a first-degree polynomial:

$$X = a_0 + a_1 x + a_2 y, Y = b_0 + b_1 x + b_2 y, \quad (3)$$

where a_i, b_i are coordinate connection parameters. Each resulting triangle has its own connection parameters, which are determined using control points uncontrollably.

Next, using the coordinates found, the coordinates are recalculated for all pixels inside the triangle. For points lying on the edges of the triangles, the coordinates are calculated twice.

This method allows to correct inaccuracies that occur at the boundaries between several images to obtain a seamless gluing [8].

All these transformations are performed on both images (a mask-with the image of a person who uses drugs and a photograph of the user/visitor). The size, the angle of inclination of each triangle of the mask is given in accordance with the triangle on the image of the user. Later, when applied, it will lead to the correspondence of the positions of the parts of the face - the nose, mouth, eyes, and so on. Artifacts associated with the size mismatch of these parts will be minimized. The schematic result is shown in the figure 4.

After performing the conversion for each triangle, they are ready to align with the user's face. In the implementation of the algorithm, the operations of transformation and insertion were performed by one method.

8. ALPHA BLENDING

The last but one operation is alpha blending. This operation is key for the entire algorithm. In the blending process, each pixel is combined with the corresponding pixel of another image with α coefficient that controls

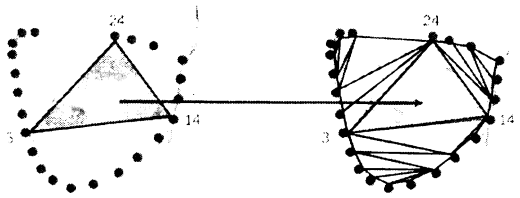

Fig 4. – Affine transformations

the pixel weight of one of the images. Selection of this coefficient for each mask was carried out manually on the basis of its individual characteristics (for example, the overly characteristic mouth shape, which cannot be correctly combined with the mouth on the user's image and avoids the appearance of artifacts on the resulting image).

Also, pre-masks were edited in a graphical editor to neutralize such characteristic features. The example in Figure 5.

Fig.5 – Original mask (left) and edited (right).

To morph image I into image J we need to first establish pixel correspondence between the two images. In other words, for every pixel (x_i, y_i) in image$I$, we need to find its corresponding pixel (x_j, y_j) in image J. That is done after affine transformation. Then we can blend the images in two steps. First, we need to calculate the location (x_m, y_m) of the pixel in the morphed image. It's given by the following equations:

$$x_m = (1 - \alpha)x_i + \alpha x_j, y_m = (1 - \alpha)y_i + \alpha y_j \quad (4)$$

Second, we need to find the intensity of the pixel at (x_m, y_m) using the following equation:

$$M(x_m, y_m) = (1 - \alpha)I(x_i, y_i) + \alpha J(x_j, y_j) \quad (5)$$

We can adjust the parameter α, thereby changing the degree of similarity of the final image to the original image $I(\alpha = 0)$ or $J(\alpha = 1)$.

The final step is automatic color correction. It is performed indirectly, using the seamlessClone method from the OpenCV library, which copes with its task quite adequately and does it very quickly.

9. RESULTS

Next, fig. 6 and 7 shows examples of how the software works on well-known and publicly available

images. As you can see, the algorithm correctly handled the tilt and turn of the head. To obtain these results, several masks with different characteristic features were used - potholes and wrinkles for the image on the left, and ulcers on the cheeks for the image on the right.

Fig.6 - Results of the program operation

Fig.7 - Results of the program operation

Some artifacts are visible on the brow-brow line. They are unavoidable for this algorithm, since automatic color correction cannot completely eliminate the difference in the range of different images, as well as skin texture and resolution of images are also different (which is also impossible to change).

10. CONCLUSION

Software has been developed that produces a combination of faces for drawing traces of illicit drug use on a person's photo. It works in real time on low-power hardware.

During the operation of the application, the CPU load is about 5-10%; components are used to display images through the GPU to speed up the display.

11. REFERENCES

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

[2] P. Viola and M.J. Jones, «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2001.

[3] R.Gonzalez, R.Woods, "Digital Image Processing", ISBN 5-94836-028-8, publishing house: Tekhnosfera, Moscow, 2005. - 1072 p.

[4] Mestetsky L. M., "Mathematical methods of pattern recognition", Moscow State University, Moscow, 2002–2004., P. 42 - 44.

[5] Yoav Freund, Robert E. Schapire, «A Short Introduction to Boosting», Shannon Laboratory, USA, 1999., pp. 771-780.

[6] Skvortsov A.V. Delaunay triangulation and its application. - Tomsk: Tomsk University Press, 2002. - 128 p.

[7] Beklemishev D.V. The course of analytic geometry and linear algebra. - Edition 12th, corrected. - M .: Fizmatlit, 2009. - 312 p.

[8] Yerahmiel Doytsher, A rubber sheeting algorithm for non-rectangular maps, Computers&Geosciences, Volume 26, Issues 9–10, 1 November 2000, Pages 1001–1010.

[9] Pérez, P., Gangnet, M., and Blake, A., "Poisson image editing," ACM Transactions on Graphics (TOG), 22(3), pp. 313–318, 2003.