REAL-TIME FACE RECOGNITION IN VIDEO STREAMING

Ivanov Ivelin, Burgas Free University, ivelin8910@gmail.com
Vanteva Violeta, Burgas Free University, v_vanteva@abv.bg
Georgieva Penka V., Burgas Free University, pgeorg@bfu.bg

Abstract: Face detection and recognition are areas in *Artificial Intelligence* that play a key role in information and people security and have various fields of application - control of groups of people, traffic monitoring, finding missing persons, searching for criminals and others.
In this study, a system for real-time face detection and recognition in video streaming is proposed, the system's model and implementation are described and test results are analyzed.

Keywords: face detection, face recognition, artificial intelligence, real-time software system

РАЗПОЗНАВАНЕ НА ЛИЦА ОТ ВИДЕО ПОТОК В РЕАЛНО ВРЕМЕ

Ивелин Иванов, Бургаски свободен университет, ivelin8910@gmail.com
Виолета Вантева, Бургаски свободен университет, v_vanteva@abv.bg
Пенка В. Георгиева, Бургаски свободен университет, pgeorg@bfu.bg

Абстракт: Засичането и разпознаването на лица са направления в *Изкуствения интелект,* които играят ключова роля за сигурността на информация и хора с разнообразни области на приложение - контрол на групи хора, наблюдение на трафик, откриване на изчезнали хора, издирване на криминално проявени лица и други.
В тази студия е предложена система за засичане и разпознаване на лица от видео поток в реално време, като са описани моделът и реализацията на системата и са анализирани резултати от проведените тестове.

Ключови думи: засичане на лица, разпознаване на лица, изкуствен интелект, софтуерни системи в реално време.

I. Въведение

Разпознаването на лица е задача за откриване на шаблони в изображения, които са получени от различни източници на информация и по тази причина имат различни характеристики.

Ако изображението се променя във времето, то входните данн представляват видео поток.

Въпреки, че човешкото лице се състои само от две кости от две кости (череп, челюст) и 44 мускула, са възможи хиляди различни изражения на лицето. Умението за разпознаване на лица е централно и водещо за функционирането на всеки индивид, защото заема най-голям дял от посветената на обработването на зрителна информация функция (около 50% от мозъчната дейност).

Съществуват различни техники за идентифициране и удостоверяване, но основен аспект на разпознаването на лица е, че то е пасивна ненатрапчива система за проверка и идентифициране на хора [1].

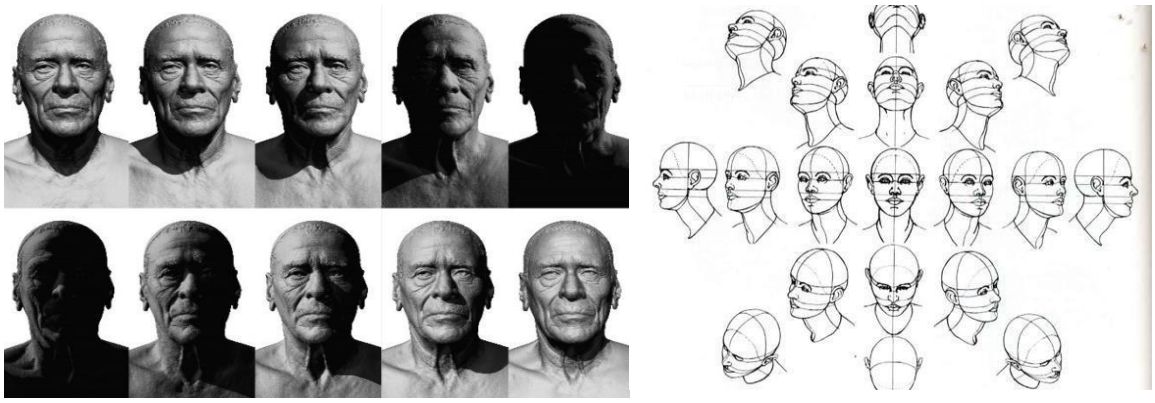
Автоматизираното разпознаване на лица започва развитието си през 60-те години на XX век. Първите полуавтоматични системи за разпознаване на лица са изисквали от администратора да открива различни отличителни белези като очи, уши, нос и уста, да изчисли разстоянията и съотношенията относно една референтна точка, след което резултатите са сравнявани с референтните налични данни [2]. През 1971 г. Голдщайн, Хармън и Леск използват 21 специфични субективни маркера (цвят на коса, дебелина на устни и др.), за автоматизиране на разпознаването [3]. Проблем на първите системи е, че както измерванията така и изчисленията са осъществявани ръчно. През 1987 г. Къбри и Сирович прилагат анализ на компонентите, което се счита за повратен момент, тъй като доказва, че са необходими по-малко от сто маркери за точно изчисление на подходящо подравнено и нормализирано изображение на лице [4]. През 1991 г. Пентланд и Търк откриват, че използването на собствени вектори може да даде възможност за откриване на лица в изображения и така да бъдат създадени надеждни автоматизирани системи за разпознаване на лица в реално време [5].

Днес технологиите за разпознаване на лица се използват за борба с паспортни измами, подкрепа на правосъдието, идентифициране на изчезнали хора, минимизиране на измамите, кражба на самоличност и други. [6]

Стандарти за изображения на лица

Съществуващите стандарти се отнасят предимно до съдържанието на изображенията и тяхното качество. За подобряване ефективността на системите за разпознаване е необходимо да се договорят стандарти, свързани с методологията на разпознаването [7]. Международният комитет за стандарти за информационните технологии (*International Committee for Information Technology Standards= INCITS*), който е признат от Организацията за развитие на стандарти за информационни технологии в САЩ (*Standards Development Organization=ANSI STO*) и Международната организация по стандартизация (*International Standards Organization=ISO*) чрез своята Международна електротехническа комисия (*International Electrotechnical Commission=IEC*) и Съвместния технически комитет (*Joint Technical Committee=JTC*) са издали 37 проекта за стандарти на биометричните данни. Международната организация за гражданска авиация е публикувала два основни документа, засягащи биометричното разгръщане на машинно четене на пътнически документи - ISO/IEC 19794-5-FDIS JTC l/SC 37, M1/04-0041 (ANSI) INCITS (385). Стандартът Ml INCITS I2 приема или пълен или каноничен образ. Изображението на лицето трябва да включва цялата глава, косата, врата и раменете и ако се съхраняват и точки на характерните черти, то те базирани на SC29/MPEG4. Съществуват специфични изисквания за позата на лицето, отнасящи се до ъгъла на завъртане, наклон и посока на завъртане, изражението, осветлението и сенките. [8] ***Предизвикателства при***

разпознаване на лица от статични изображения Съществуват разнообразни варианти на изображение на човешкото лице, тъй като то зависи от фактори като поза, осветеност и изражение, от размера на лещата на обектива, както и от времето на излагане и отклонението на камерата. На фигура 1 са показани два от основните проблеми: различното осветление (фиг. 1а) и различен наклон на лицето (фиг. 1б). От друга страна, понякога се появяват малки междуличностни вариации (пр. идентични близнаци).



Фигура 1. Съществуващи предизвикателства в областта:
а) различната осветеност на лицето; б) различен наклон на лицето.

За много от съществуващите системи за разпознаване предизвикателствата са големи и решени само до определена степен. Например, един подход за първоначално филтриране е налагането на ограничения в процеса на придобиване на изображения. Но има и случаи, в които вариациите сред изображенията на едно и също лице са по-големи от вариациите между изображенията на две различни лица, и тогава трябва да бъдат придобити по-прецизни изображения.

Предизвикателства при разпознаване на лица от видео поток Допълнителните предизвикателства при разпознаване на лица от видео поток са следните:

1) претрупан фон с много ненужни детайли;
2) обектът за разпознаване липсва за даден период от време;
3) движението на обекта може да не е успоредно на оптичната ос;
4) локализират се множество обекти;
5) различни ориентации на лицето на обекта;
6) загуба на детайлите на лицето поради размазване, причинено от движение на обекта спрямо камерата;
7) загуби, причинени от компресия на видео потока;
8) възникване на оптично изкривяване при изместване на обекта от оптичната ос и/или силно намалено разстояние до камерата.

Видове разпознаването на лица

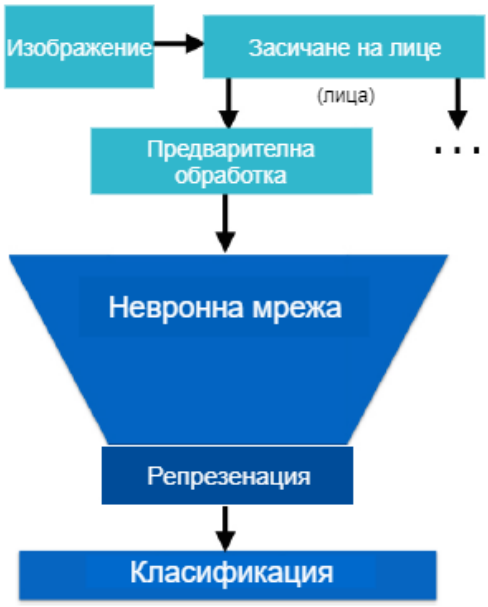
Според целите на разпознаването има три основни вида разпознаване на лица.

1. Проверка и/или удостоверяване е задача, изискваща действие на потребителя под формата на заявка за самоличност, като въпросът е *Аз ли съм този, който казвам, че съм?*, а търсенето от вида *1:1*. Тестът за проверка се извършва чрез разделяне на лицата на две групи:
   - *клиенти* – хора, които се опитват да получат достъп, използвайки собствената си самоличност;
   - *измамници* –хора, които се опитват да получат достъп, използвайки фалшива идентичност, т. е. идентичност, позната на системата, но която не е тяхна.
2. Идентификация и/или разпознаване е задача, която не изисква взаимодействие с потребителя с въпрос *Кой съм аз?*. В този случай търсенето е от вида *1:n*, а идентификационният тест се основава на предположението, че всички образи са на вече известни личности.
3. Наблюдение е задача, която е обобщение на идентификационната задача с включване на неизвестни хора чрез търсене *1:n*.

Модули на система за разпознаване

Системите за разпознаване на лица имат три концептуални модула (фиг. 2):

1) модул за засичане на лице, в който се осъществява локализация в изображението чрез шаблони;



Фигура 2. Основни модули на система за разпознаване

2) модул за предварителна обработка, съдържащ нормализация на лицето, подравняване, преобразуване, завъртане, мащабиране, корекция на светлината и други. В този модул се извличат характеристиките на лицето като компактен

набор от дискриминационни геометрични и/или фотометрични характеристики на лицето чрез методи като: анализ на главните компоненти (*PCA*) [9], линеен дискриминационен анализ на Фишер (*FLDA*), прогнози за запазване на местоположението (*LLP*), невронни мрежи и други;

3) модул за класификация след сравняване на характеристичния вектор за съвпадение с лицеви изображения, вече записани в база данни с алгоритми като : най-близкия съсед, изкуствени невронни мрежи и други.

Съществуващи решения за разпознаване на лица

През последните 10 години са създадени множество разнообразни софтуерни системи за засичане и разпознаване на лица с различни приложения, но малко от тях работят в реално време. Популярни такива системи са: *digiKam, FaceVACSVideoScan, Adobe Photoshop Lightroom, Picasa, DeepFace, FBI Sinister Next Generation Identification, Examiner, MFLO, NeoFace® Watch.*

Системи, работещи в реално време

Софтуерните системи, работещи в реално време стават все по-важни за функционирането на съвременното общество. Такива системи се използват за управление на въздушното движение, за разпознаване на обекти, за управление на командите, мрежови мултимедийни системи и други. При системите в реално време точността на поведението на системата зависи не само от логическите резултати от изчисленията, но и от физическия момент, в който се получават тези резултати [10]. Компютърна система, работеща в реално време, трябва да реагира на изменения в средата в рамките на предварително дефинирани интервали от време. Моментът, в който се генерира резултат, се нарича краен срок. Специално внимание се отделя на системите в строго реално време и гъвкаво реално време. Пропуснатият краен срок в такива системи е катастрофален и може да доведе до значителни загуби [11].

II. Модел на приложение за засичане и разпознаване на образи в реално време

Обработка на видео поток

В описаната в тази студия софтверна система за засичане и разпознаване на лица в реално време, първоначално видео материалът се разделя на кадри, като на всеки 200 милисекунди се зарежда нов кадър от съответния видео материал и той бива обработен. Такъв интервал на бързодействие е избран, за да е съвместим с времето, за което нервен импулс стига от кожата на човек до мозъка (100 милисекунди) и времето за мигане (100-400 милисекунди), според базата данни на Харвард за стойности на биологични процеси [12].

Клъстеризация

Клъстеризацията на процеса на обработка на видео кадрите, използвана при реализирането на предложения в тази студия модел, има за цел оптимизация с цел намаляване времето за обработка на кадри и съответно увеличаване бързодействието на цялата система за засичане и разпознаване на обекти. Клъстерната архитектура за поддържане на голям брой потребители балансира натоварването, като всяка заявка е насочена към конкретен възел. [13]

Логическо описание на приложението

Приложението за разпознаване на лица се състои от четири основни модула (фиг. 3):

1) предварителна обработка на постъпващия видео материал;
2) клъстеризация на програмата;
3) засичане на образ в отделните кадри;
4) разпознаване на съответното засечено лице, ако такова е намерено.

Фигура 3. Логически модел на приложението

1. *Предварителна обработка на постъпващия видео материал*

След стартиране на приложението се осъществява връзка с камерата на всеки 200 милисекунди за приемане на кадри.

Всеки постъпил кадър се преобразува от цветно в черно-бяло изображение.

След това преобразование, кадърът се предава на следващия модул.

2. *Клъстеризация на програмата*

Софтуерното приложение асинхронно подава изображение на един от възлите на всеки 200 милисекунди, като решението избор на възел се взима в реално време от самата програма.

Всеки възел обработват конкретното изображение за 50-200 милисекунди. Броят на възлите на клъстера се определя от натовареността на системата в даден момент.

Основно предимство на разпаралеляването на тази част от алгоритъма за разпознаване е, че системата има възможност да обработва кадри, едновременно постъпващи от голям брой различни машини или устройства. След изпращането на

кадър до съответен възел, управлението се поема от следващия модул на приложението.

3. *Засичане на образ в отделните кадри*

Компонентът „засичане на образ“ се реализира отделно на всеки възел.

Съществуват различни алгоритми за засичане на обекти в изображение, като методите за засичане на лица са основно два вида:

намиране на лица в изображения с контролиран фон;

намиране на лица в изображения без ограничения за средата. При първия подход се използват изображения с обикновен монохроматичен фон или такъв, който е предварително зададен. Това улеснява премахването на фона, като след това остават само границите на самото лице, ако такова присъства в изображението. Вторият подход при засичането на лица е засичане без наличие на ограничения в средата. Използват се алгоритми, базирани на геометрични модели, разстояние на Хаусдорф [14] и други.

В настоящата система за засичане и разпознаване на лица е реализиран алгоритъмът на Виола-Джоунс [15], който се състои от четири етапа:

1. избор на характеристики на Хаар;
2. създаване на интегрално изображение;
3. обучение с *Адабууст;*
4. създаване на ефективни каскадни класификатори.

Този метод за засичане на лица използва зависимости, присъстващи в човешките лица, които се представят чрез черни и бели правоъгълни области. Тези правоъгълни характеристики се изчисляват чрез интегрално изображени $ii(x,y)$, при което на мястото на пиксел с координати (x,y) се записва сумата от пикселите, които се намират над и вляво от него:

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y') \ , \qquad (1)$$

където $i(x,y)$е оригиналното изображение.

Интегралното изображение се пресмята с едно обхождане на оригиналното изображение със следната двойка повторения:

$$s(x,y) = s(x,y-1) + i(x,y) \qquad (2)$$

$$ii(x,y) = ii\,(x-1,y) + s(x,y)\,, \qquad (3)$$

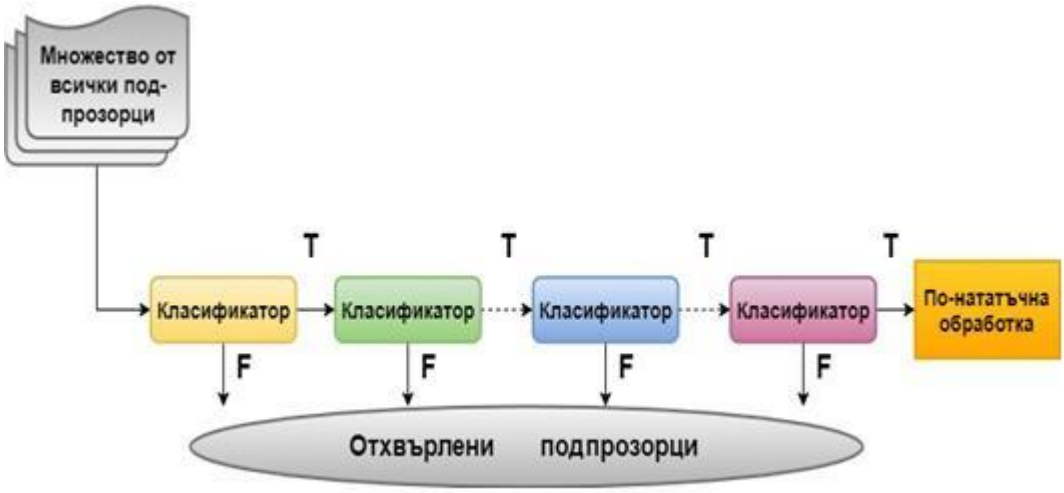
при $s(x,-1) = 0$ и $ii\,(-1,y) = 0$.

Алгоритъмът за обучение *Адабууст* повишава класификационната способност на прост алгоритъм за обучение [16]. Алгоритъмът за обучение на еднослойна невронна мрежа претърсва множество от неврони и открива този с най-голяма класификационна грешка. За подсилване на обучението му се използва функция $h(x,f,p,\theta)$, дефинирана по следния начин:

$$h(x,f,p,\theta) = \begin{cases} 1, & \text{при} \quad p.f(x) < p.\theta \\ 0, & \text{навсякъде другаде} \end{cases} \qquad (4)$$

където f е елемент, p - полярност и θ – праг.

В последния етап на алгоритъма Виола-Джоунс се изгражда каскада от класификатори за намалявана на времето и повишаване на ефективността при засичане. Първо се прилагат по-простите класификатори с цел отхвърляне на поголямата част от подпрозорците, след което се използват по-сложните класификатори. Каскадата се изгражда от обучаващи класификатори отново чрез обучение *Адабууст*. Общата форма на процеса на засичане е дърво за вземане на решения, което в случая се нарича „каскада“. Положителен резултат от първия класификатор задейства изпълнението на втори класификатор, който също бива коригиран, за да постигне по-високи нива на засичане. Положителният резултат от втория класификатор задейства трети такъв и така до края на каскадата. Негативният резултат във всеки момент води до незабавното отхвърляне на текущия, установен като негативен подпрозорец. Този процес е илюстриран на фигура 4.



Фигура 4. Принцип на пораждане на класификатори и отхвърляне на подпрозорци

Използването на каскадна структурата дава възможност за отхвърляне на възможно най-много подпрозорци във възможно най-ранна фаза. Фалшиво-положителната величина F на каскадата се изчислява по формулата:

$$F = \prod_{i=1}^{K} f_i \ , \tag{5}$$

където K е броят на класификаторите и f_i е фалшиво-положителната величина на iтия класификатор на примерите, които достигат до него. Коефициентът на откриване D е следният:

$$D = \prod_{i=1}^{K} d_i \ , \tag{6}$$

където K е броят на класификаторите и d_i е степента на засичане на i-тия класификатор на примерите, които достигат до него.

Като се имат предвид конкретните цели за общите фалшиво-положителни стойности и коефициентите на засичане, целевите нива могат да бъдат определени за всеки етап

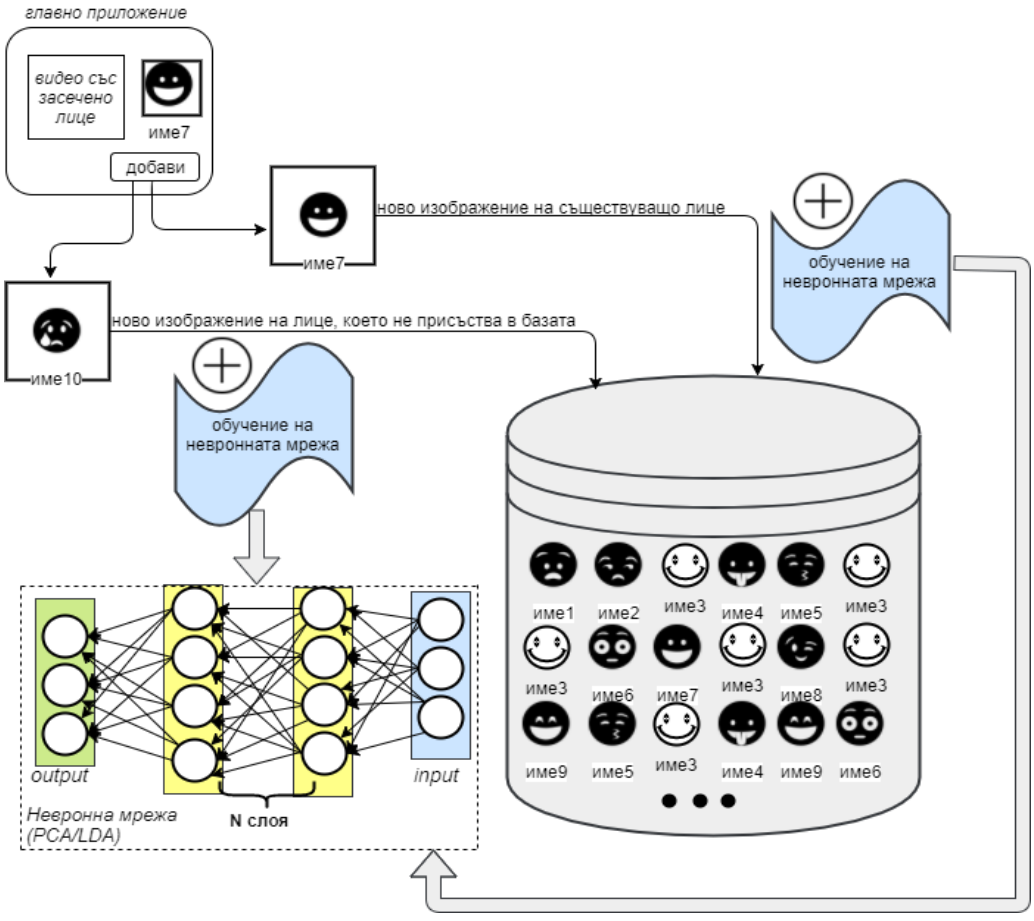
от каскадния процес. Например, коефициент на засичане *0,9* може да бъде постигнат чрез *10*- етапен класификатор, ако всеки етап има степен на откриване от *0,99*.
Броят на характеристиките, оценени при обработване на реални изображения е вероятностен процес. Очакваният брой характеристики *N*, които се оценяват е:

$$N = n_0 + \sum_{i=1}^{K} \left(n_i \prod_{j<i} p_j \right), \tag{7}$$

където K е броят на класификаторите, p_i е положителната скорост на i-тия класификатор и n_i е броят на характеристиките в i-тия класификатор.

4. Разпознаване

В този модул на приложението се постигат няколко основни задачи. Първо се съставя база от изображения в реално време, като след засичането на лице се предоставя възможност потребителят да въведе името му и това лице автоматично се добавя в базата от данни с тестови изображения (фиг. 5). След това невронната мрежа се обучава над новополучената обучителна база, като това се осъществява едновременно със засичането. Алгоритмите, които са използвани в невронната мрежа имат за основна цел да премахнат общите черти и да сравняват засеченото лице с уникалните черти с тези, които са вече разпознавани от системата. Това се постига с *Анализ на главните компоненти* (*АГК*) [17].



Фигура 5. Принцип на работа на модула за разпознаване

Анализ на главните компоненти (*АГК*)

Нека в базата от данни има n на брой обекта, като за всеки от тях се наблюдават m на брой характеристики. Тези наблюдения се записват в базата от данни като n на брой m-мерни вектори $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n$. По този начин всички наблюдениия могат да се разглеждат като матрица X от тип $m \times n$. В *АГК* се използва следната теорема.

Теорема. (спектрална теорема) Ако за матрицата A от тип $n \times n$ е изпълнено $A = A^T$, то:

1) всички собствени стойности $\lambda_1, \dots, \lambda_n$ на матрицата A са реални числа и съществуват ненулеви собствени вектори $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n$ с реални координати, такива че $A.\vec{v}_j = \lambda_j.\vec{v}_i$,за всяко $j = 1, 2, \dots n$;
2) собствените вектори, съответстващи на различни собствени стойности са ортогонални;
3) съществува диагонална матрица $D = diag(\lambda_1, \dots, \lambda_n)$ от тип $n \times n$ и ортогонормална матрица U от тип $n \times n$ с елементи по стълбове съответните координати на собствените вектори, за които е изпълнено следното равенство:

$$A = U.D.U^T$$

В системата за разпознаване тази теорема се прилага върху симетричните матрици $X^T.X$ от тип $n \times n$ и $X.X^T$ от тип $m \times m$. Съществува връзка между собствените стойности и собствените вектори на $X^T.X$ и $X.X^T$. Наистина, нека $\vec{v}$ е ненулев собствен вектор на матрицата $X^T.X$, съответстващ на собствената стойност $\lambda \neq 0$. Тогава е в сила равенството:

$$(X^T.X).\vec{v} = \lambda.\vec{v}\,, \tag{8}$$

откъдето след умножаване отляво с X и прилагане на свойствата за умножение на матрици се получава равеството:

$$X.X^T.(X.\vec{v}) = \lambda.(X.\vec{v}) \tag{9}$$

От (9) следва, че $X.\vec{v}$ е собствен вектор на матрицата $X.X^T$, а λ е съответната собствена стойност. Доказателството за $X.X^T$ е аналогично. [18]

Доказаното свойство е особено важно в случай, че Това предположение, което бе и доказано, е доста важно в случая, в който стойностите на m и n се различават значително (пр. $m = 500$ и $n = 2$).

Друго важно свойство, което се използва в системата за разпознаване е фактът, че собствените стойности на матриците $X^T.X$ и $X.X^T$ са неотрицателни числа. Наистина,

нека $\vec{v}$ е ненулев собствен вектор на матрицата $X^T . X$, съответстващ на собствената стойност $\lambda \neq 0$. Тогава за дължината на вектора $X.\vec{v}$ е в сила:

$$|X.\vec{v}|^2 = (X.\vec{v})^T.(X.\vec{v}) = \vec{v}^T.(X^T.X).\vec{v} = \vec{v}^T.(\lambda.\vec{v}) = \lambda.(\vec{v}^T.\vec{v}) = \lambda.|\vec{v}|^2 \geq 0, \quad (10)$$

Следователно $\lambda \geq 0$.

Нека векторът $\vec{\mu}$ е средноаритметично на векторите $\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n$:

$$\vec{\mu} = \frac{1}{n}(\vec{x}_1 + \vec{x}_2 + \dots + \vec{x}_n), \quad (11)$$

вариацията $\mathrm{Var(X)}$ е:

$$\mathrm{Var(X)} = \frac{1}{n-1}((\vec{x}_1 - \vec{\mu})^2 + (\vec{x}_2 - \vec{\mu})^2 + \cdots + (\vec{x}_n - \vec{\mu})^2) \quad (12)$$

и матрицата Y от тип $m \times n$ има за j –ти стъб вектора $\vec{x}_j - \vec{\mu}$ за $j = 1,2,\dots,n$.

Конструира се матрица S от тип $m \times m$ по следния начин:

$$S = \frac{1}{n-1} Y.Y^T, \quad (13)$$

за чиито елементи S_{kj} е изпълнено:

$$S_{kj} = \begin{cases} \text{вариацията на } k-\text{тата характеристика}, & \text{при } k = j \\ \text{ковариацията на } k-\text{тата и } j-\text{тата характеристики}, & \text{при } k \neq j. \end{cases}$$

От (13) следва, че матрицата S е симетрична и нека $\vec{u}_1, \vec{u}_2, \dots, \vec{u}_m$ са собствените й вектори, отговарящи на собствените й стойности, предварително подредени в низходящ ред $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$. Тези собствени вектори се наричат *главни компоненти* на X. От една страна следата T на матрицата S е равна на сумата вариациите, а от друга – на сумата от нейните собствени стойности ($T = \lambda_1 + \lambda_2 + \lambda_3 + \cdots + \lambda_m$).

Тогава:

- посоката на първия главен компонент $\vec{u}_1$ отчита значимостта на λ_1 в общата вариация, равна на $\frac{\lambda_1}{T}$; посоката на втория главен компонент $\vec{u}_2$ отчита значимостта $\frac{\lambda_2}{T}$на λ_2 и т.н.;
- от $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$ следва, че векторът $\vec{u}_1$ посочва най-значимата компонента;
- от всички ортогонални на $\vec{u}_1$ направления, векторът $\vec{u}_2$ посочва найзначимата компонента;
- от всички ортогонални на $\vec{u}_1$ и $\vec{u}_2$ аправления, векторът $\vec{u}_3$ посочва найзначимата компонента и т.н.

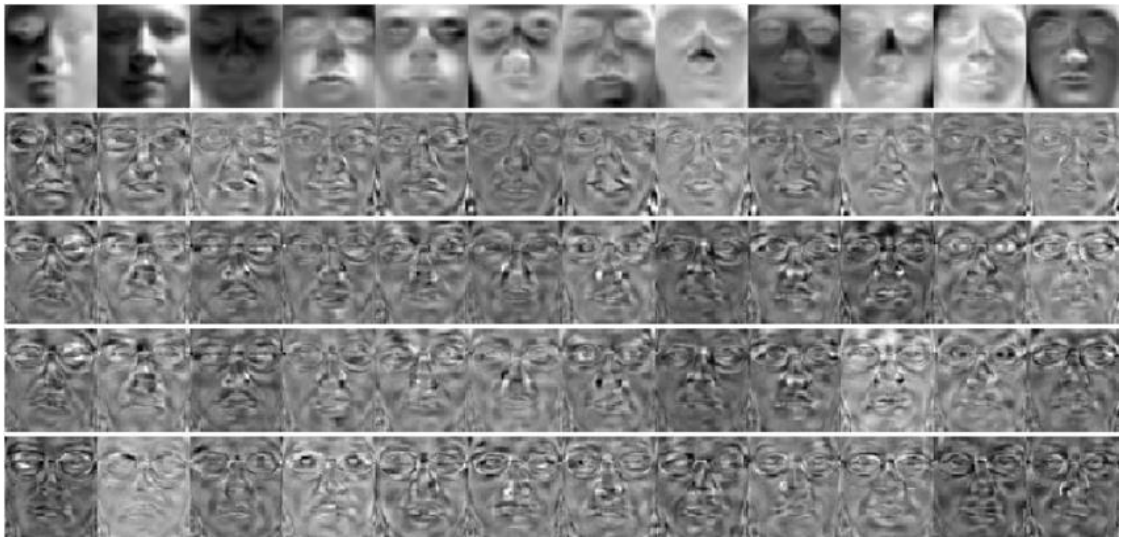
В приложенията, използващи *АГК,* първите собствените стойности на S се получават значително по-големи от всички останали. Така се оказва, че посоките примерно на първата и втората главна компонента обясняват *почти всички* вариации в данните.

Така, дори и да се отчитат много на брой характеристики, използването на двуе напълно достатъчно за получаване на състоятелни резултати.

Алгоритъмът на *АГК* се прилага в следните три основни стъпки:

1. събират се n на брой проби с m −мерни данни$\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n$, изчислява се средната μ, конструират се матриците Y и S,
2. пресмятат се собствените стойности на S, подредени в низходящ ред $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$ и съответните собствени вектори$\vec{u}_1, \vec{u}_2, \dots, \vec{u}_m$,
3. анализират се получените резултати, като се отговаря на следните въпроси: дали малък брой от λ_i са много по-големи от всички останали, възможно ли е намаляване на размера на съществуващите данни, кои от променливите са най-важни и други.

Методът на главните компоненти е широко използван в разпознаването на лица, поради оптимизацията, която може да бъде постигната. На фигура 6 е показан примерен резултат, получен за обучителната база от изображения след прилагането на анализа на главните компоненти. За лицето на всеки човек отбазата са останали само част от чертите, като това са само уникалните му характеристики. В случая първите главни компоненти са много по-значими от всички следващи.



Фигура 6. Обучителна база след прилагането на АГК

Невронната мрежа, използвана в конкретното рабти идентично. След като невронната мрежа достигне до разпознаване, тя връща като изход име на разпознатото лице, ако е разпознато такова, заедно с вероятност, показващ степента на сигурност. Тази информация се изпраща пакетирана към главното приложение, което на базата на резултата, може да взема различни решения и също изписва в самия видео материал на екрана името на човека, който е разпознат, заедно с рамката, която е подадена като правоъгълник за мястото, на което е засечено лице.

III. Реализация на модела

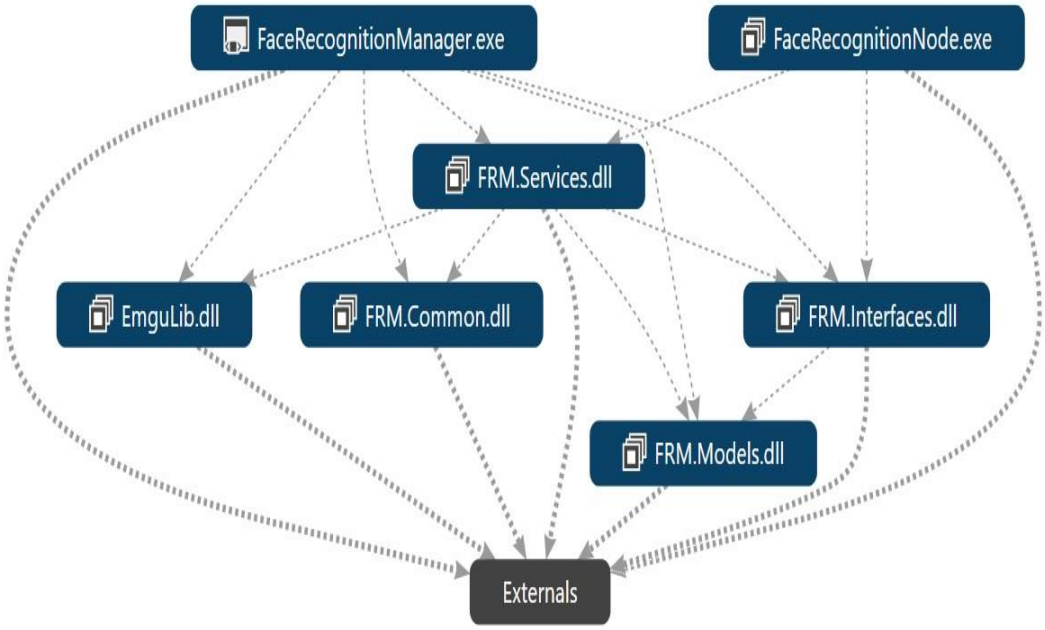
Използвани технологии за софтуерната реализация

Разработката на система за разпознаване на лица включва софтуерно приложение, реализирано на езика за програмиране от високо ниво C# и платформата .NET.
Изборът на C# и .NET е обусловен от наличието на библиотеката с отворен код OpenCV (Open Source Computer Vision), подходяща за системи за засичане и разпознаване на лица. OpenCV е библиотека от функции за програмиране, насочени главно към компютърното възприятие в реално време с компоненти като машинно обучение, дървета на решения, спускане по градиента, алгоритъм *k*-ти най-близък съсед, поддържаща векторна машина (SVM), дълбоки невронни мрежи (DNN) и други.

Структура на приложението

Обща схема на приложението е показана на фигура 7. То се състои от два основни компонента:

1) **FaceRecognitionManager.exe** за вземане на кадри от източника тип „камера“ и изпращането им към подчинените му възли;
2) **FaceRecognitionNode.exe** за разпознаването на лице в съответния подаден кадър.



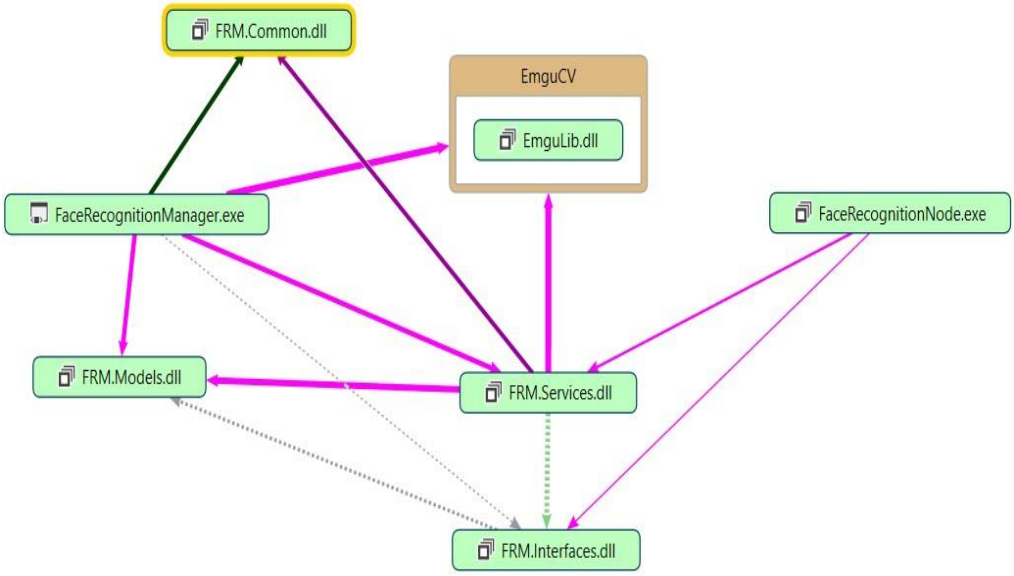
Фигура 7. Диаграма на връзките и проектите на системата за разпознаване

Архитектурата на приложението е многослойна:

- **FRM.Models.dll** съдържа всички нужни класове на приложението;

- **FRM.Common.dll** съдържа всички общи помощни функции;
- **FRM.EmguLib.dll** съдържа всички алгоритми, нужни за обработката и засичането на лице от снимка;
- **FRM.Interfaces.dll** съдържа всички интерфейси на класовете, като по този начин се осигурява независимост на главното приложение от конкретната имплементация и смаяната й, докато приложението е в работен режим „runtime“;
- **FRM.Services.dll** е библиотека, в която се съдържа цялостната логика нужна за създаване и поддържане на комуникацията между главното приложение и неговите възли.

EmguCV е Dll файл (динамична библиотека), представляващ интерпретация на библиотеката **OpenCV** в средата .NET на езика C#. Към **EmguCV** се обръщат приложенията **FaceRecognitionManager.exe** и **FRM.Services.dll** (фиг. 8).

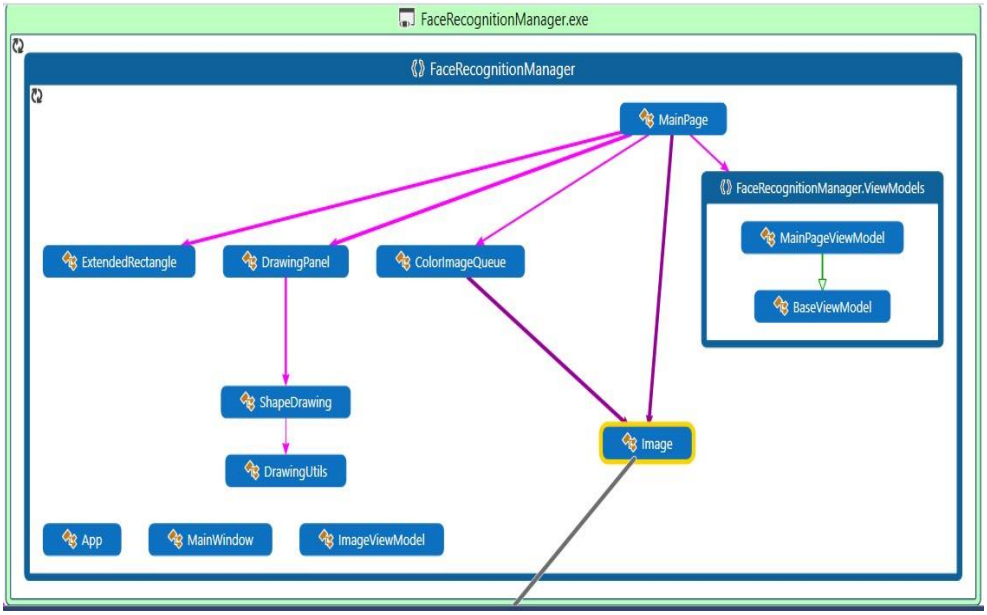


Фигура 8. Диаграма на приложенията в системата и връзките между тях и библиотеката с отворен код EmguCV

На фигура 9 е показана клас-диаграмата на главното приложение с име **FaceRecognitionManager**. Това приложение е изпълнима библиотека и се състои от няколко класа, служещи за визуализация резултатите на екрана:

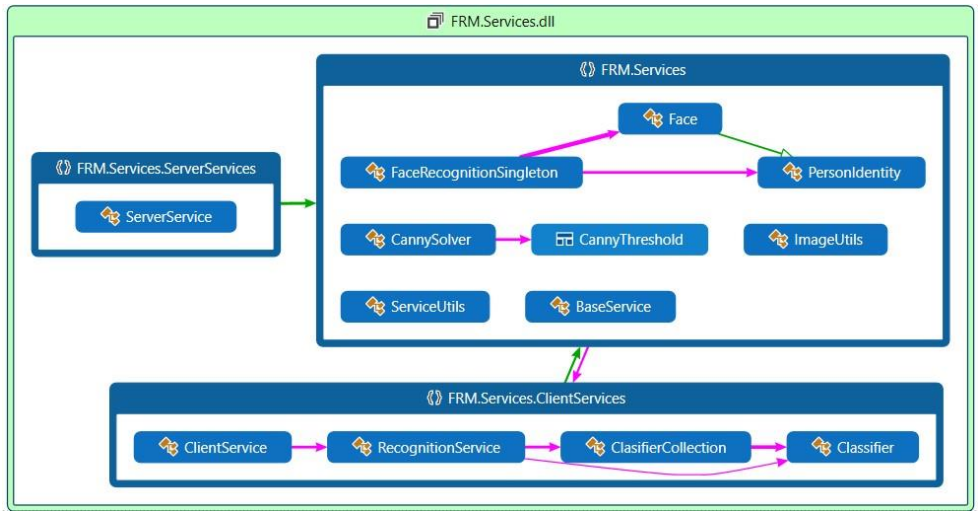
- **ExtendedRectangle** съдържа координатите на подаден правоъгълник, както и информация за типа му, определен от това дали разпознатият правоъгълник е на лице, око, уста и т.н.,
- Класовете **App**, **MainWindow** репрезентират съответното приложение и прозореца, който то създава, в който се добавят другите компоненти на интерфейса,

- **ShapeDrawing**, **DrawingUtils** са класове, съдържащи функции за рисуване на форми, текст и изображения на екрана,
- **DrawingPanel** – контролен панел, съдържащ нарисуваното изображение, като използва функционалностите предоставени от класа **ShapeDrawing**,
- **Image** – клас, съхраняващ модела на изображението в различните му варианти - цветно и черно-бяло, които се съхраняват и разглеждат като едно цяло,
- **ColorImageQueue** –клас за реализиране на циклична опашка с даден капацитет. Това нужно, поради това че процесът по вземане на кадри от устройството “камера“ и процесът по изпращане на кадри за разпознаване са асинхронни. Така единият процес добавя елементи към опашката, а другият ги изважда и изпраща за разпознаване. В обработката на изображения в реално време се позволява даден кадър да се отхвърли, поради това че той вече е нерелевантен в даденият период от време т.е. не е нужно да обработваме стари кадри, които не отговарят на реалното състояние на средата в момента,
- **MainPage** – класът е репрезентация на страницата в приложението, в която са подредени всички визуални контроли. Реализиран е посредством MVVM модел, което е подход в софтуерните архитектури, осигуряващ двупосочна комуникация между данните и интерфейса. Така, ако някоя променлива си промени състоянието или стойността, това веднага се отразява на екрана, и обратно, ако състоянието на интерфейса се промени от взаимодействието на потребителя с него, то това ще промени стойностите на променливите в модела,
- **MainPageViewModel** съхранява всички данни, нужни за взаимодействието на потребителя с главната страница в приложението.



Фигура 9. Клас- диаграма на проекта FaceRecognitionManager, съдържащ главната функционалност

Следващият реализиран проект **FRM.Services** е главна част от реализираната система за разпознаване. Неговата структура е представена подробно на фигура 10. Тази библиотека се използва и в двете споменати вече приложения. Състои се главно от три namespace-а: един, използван в частта на сървъра/главния нод, един за частта на подчинените нодове/възли и един общ: *Клиентски клас, Сървърни класове*.



Фигура 10. Клас-диаграма на проекта Services

Реализираната многослойна архитектура на системата е необходима, за да се получи не само изрядност на кода, но и за да се реализират самостоятелно отделните логически модули.

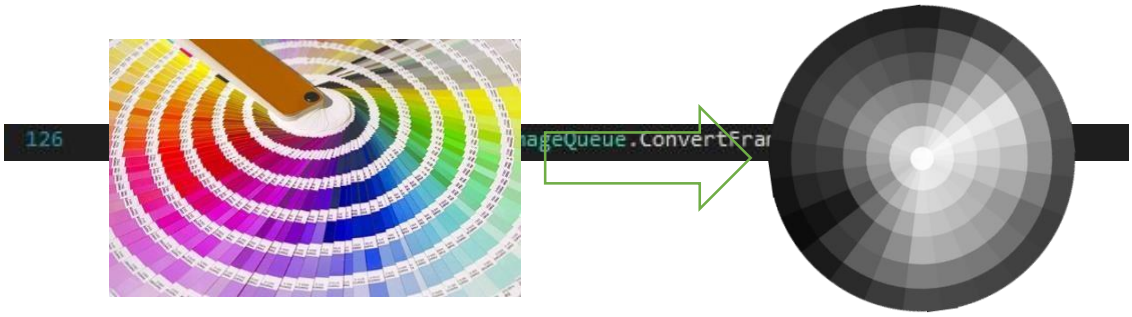
Осъществяване на комуникация с устройство за видео поток Първи необходим инструмент за да съществува системата за разпознаване, е източник на видео поток в реално време. Видео потокът може да бъде получен от камера на лаптоп или друга външна камера, като комуникацията със самото устройство се осъществява веднага при стартиране на приложението. Това е реализирано с DirectX пакета, който предоставя възможности за конструиране на графики на монитора и за получаване на изображения от външен източник (пр. камера), свързан с компютъра, както и за звук други. При обръщане към DirectX се създава канал за комуникация на камерата с приложението. Това се реализира чрез Windows драйверите на съответното устройство, като към тях се обръща именно DirectX. След осъществяване на въпросната комуникация, тя получава свой идентификационен номер в приложението, който се използва като обръщение към устройството (в случая камерата, вградена в компютъра). Чрез този идентификационен номер могат да се изпращат заявки към камерата в реално време от нея. След получаването на кадъра, той се визуализира в прозорец на приложениетосъщо в реално време. По този начин човешкото око възприема потока от кадри като непрекъснат видео стрийм в реално време.
Програменият код, който реализира прихващането на кадъра от камера е:

```
1 reference
private void Tick()
{
    var frame = capture.QueryFrame();
```

ICaputure е интерфейс, който подпомага обръщението към камерата, като предоставя услуга за прихващане на кадър/изображение от видео поток.

Преобразуване на цветни изображения в черно-бели

Всеки кадър от видео потока се подлага на процедура по преобразуване на пикселите от цветната скала в черно-бяла (фиг. 11). На всеки пиксел от изображението съответства наредена тройнка стойности, съответващи на интензитета на червеното, зеленото и синьото от RGB скалата. Тези получени стойности се умножават със съответен коефициент, след което получените числа се сумират. От получената стойност се определя интензитета на сивото в интервала [0,255], като ако нула отговаря на черен цвят, а за 255 – на бял. От получените стойности се конструира RGB обект, който представлява полученият черно-бял цвят на съответния пиксел. По този начин се получава един цвят от трите интензитета и след като този алгоритъм се приложи върху цялото изображение, се получава чернобял кадър.



Фигура 11. Преобразуване от цветна в черно-бяла скала

Софтуерната реализация на тази функционалност се осъществява чрез извикване на функция, реализирана в класа **ColorImageQueue**. Програмният фрагмент за извикване на функция за цветово преобразуване е:

Преобразуването между двете скали се получа по формулата:

$$RedPixel * 0.3 + GreenPixel * 0.59 + BluePixel * 0.11,$$

Като полученият резултат се подава като параметър и по този начин се получава изображение, което има нюанси на сивото сивото.

Клъстеризация с цел оптимизиране

Клъстерът от възли в приложението е разработен с помощта на .NET технологията за уеб базирани функции WCF. Тази технология позволява напълно дуплексна асинхронна комуникация между главното приложение и отделните възли. Комуникацията се осъществява посредством TCP/IP протокол, който предоставя механизъм за сигурност и консистентно предаване на информацията, преминаваща през него. Протоколът предоставя и бързина спрямо алтернативните протоколи за комуникация.

Процес по засичане на лице в даден кадър

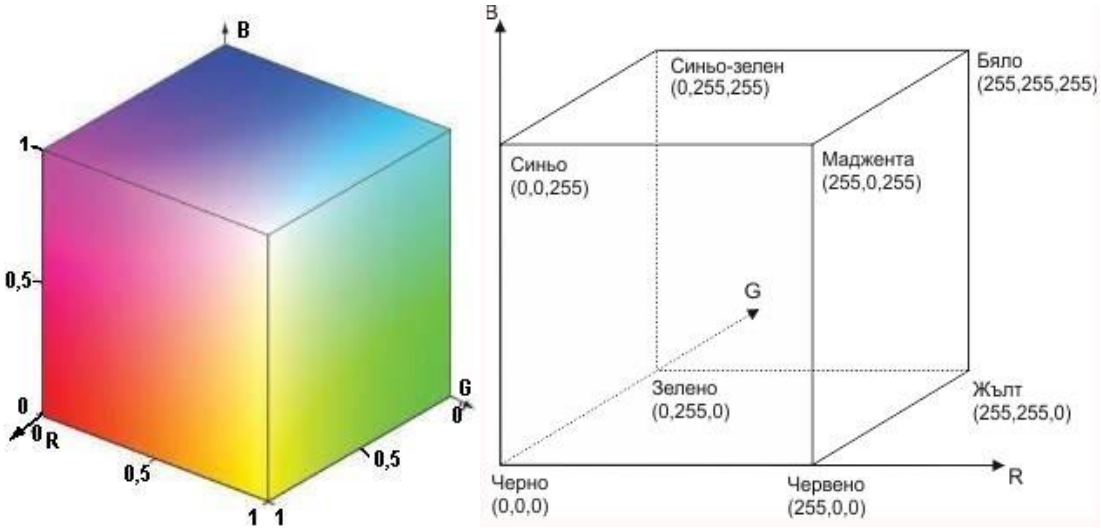
Процесът на засичането се състои главно в това да се изпрати извлечният и обработен предварително кадър на невронната мрежа, която реализира алгоритъма Виола-Джоунс. В приложението има различни опции за засичане. Те са свързани с това, че използваната библиотека предоставя намирането на различни характеристики на човешкото лице като функционалност. Включени са дескриптори:

- ✓ за засичане на очи,
- ✓ за наличие на очила,
- ✓ за засичане на ляво око,
- ✓ за засичане на дясно око,
- ✓ за засичане на лице анфас,
- ✓ за засичане на лице в профил,
- ✓ за засичане на цяло тяло,
- ✓ за засичане на долна част на тялото, ✓ за засичане на горна част на тяло, ✓ за засичане на усмивка.

Тези дескриптори са описани посредством XML файлове, които служат за йерархично структуриране на данни. Дескрипторът е описателен файл, който съдържа строго йерархично структурирани числа, които подпомагат действието на невронната мрежа. Преди самият взет вече кадър да се предаде на процеса за засичане, се прави проверка на това дали има клиенти. Ако все още има клиент, то кадърът се изпраща за засичане. Ако клиентът вече не използва в приложението, то текущият кадър се премахва от опашката с чакащи за проверка кадри, защото вече е ненужен.

Конструиране и поддържане на базата данни

Базата данни от изображения се съхранява във файлова система, като всички изображения автоматично се структурират по папки с имената на разпознатите хора за улесняване на повторно зареждане на изображенията и обучаване на алгоритъма за разпознаване на лица. Изображенията се пазят във формат PNG (Portable Network Graphics), с възможност за 16,7 милиона цвята на всеки пиксел (фиг. 12) и прозрачност, която се предоставя от четвърти слой.



Фигура 12. 3D RGB куб с цветови координати

Процес по разпознаване на засечено лице

Преди разпознаването на лицето, системата е осъществила засичането на лице и обучението на невронната мрежа, при наличието на ново лице в нея. Процесът по разпознаване се състои в избор на кой алгоритъм за разпознаване, като съответните данни се подават на невронната мрежа. Проверява се кой клиент е изпратил заявка за разпознаване на съответния кадър и дали има вече разпознато лице от всички засечени в кадъра. Накрая се изписва името на разпознатото лице на съответния кадър. Това име се запазва в променлива ‚*name*‘, която след това се присвоява като характеристика на четириъгълник, съдържащ и координатите на намереното и разпознато лице. В края на разпознаването текущият четириъгълник се изважда от опашката с неразпознати все още, но засечени обекти.

За да се прекрати действието на системата към невронната мрежа, се извиква метод Predict, реализиран в класа-модел **FaceRecognitionSingleton**.

IV. РЕЗУЛТАТИ ОТ ИЗСЛЕДВАНЕТО

Използвани технологии за провеждане на тестове

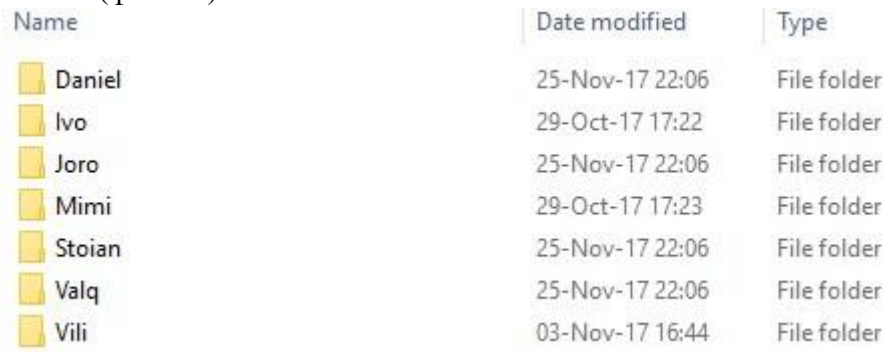
Тестовете са направени на машина с процесор Intel i7 четвърта генерация 2.5GHz, като е симулирана връзка между много машини в клъстерната система. Параметрите на използваната камера са ключови за тестовете и резултатите, като всички тестове са проведени при следните характеристики:

- 60 градуса на полезрение;
- фиксиран фокус;
- 4мм фокално разстояние -;
- 30 кадъра/секунда при 640x480 резолюция.

В следващи тестове качеството на изображенията са намалени дори повече с цел да се покаже, че работоспособността на системата е в известна степен почти независима от параметрите на устройството.

Файлова база от данни

В първоначалната версия на приложението базата данни от изображения се описва с XML файлове, които представляват йерархична структура, която е лесна за четене от приложение, но все пак трудоемка за създаване. В текущата разработка е постигната елементарната структура на файловата база от данни. Изображенията се съхраняват в папки, като всяка папка се кръщава с името на човека, на когото принадлежат дадените снимки (фиг. 13).



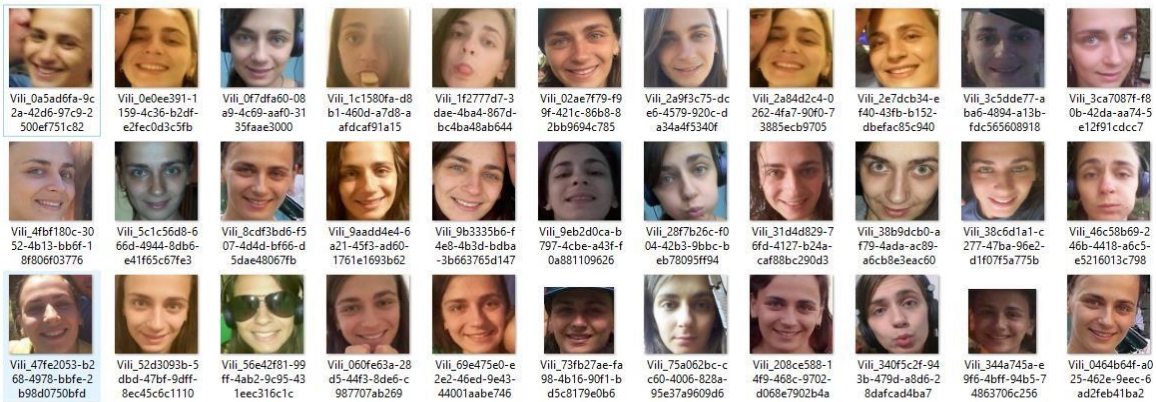
Фигура 13. Структура на файловата база данни

Цялата процедура по добавяне/премахване на папки и снимки е изцяло автоматизирана, т.е. се изпълнява от системата при необходимост.

В интерфейса на приложението са налични снимките, които се засичат от системата, заедно с поле за въвеждане на име на съответния засечен във видео потока човек. Това предоставя възможност за добавяне на нови изображения и хора в реално време, като допълнителни забавяния няма.

След всяко добавяне, невронната мрежа трябва да се обучи отново.

Приложението запазва вече изрязаните лица във формат .png с цел постигане максимално точно качество. Имената на теи изображения се конструират от системата, така че да бъдат използвани в процеса на разпознаване (фиг. 14).

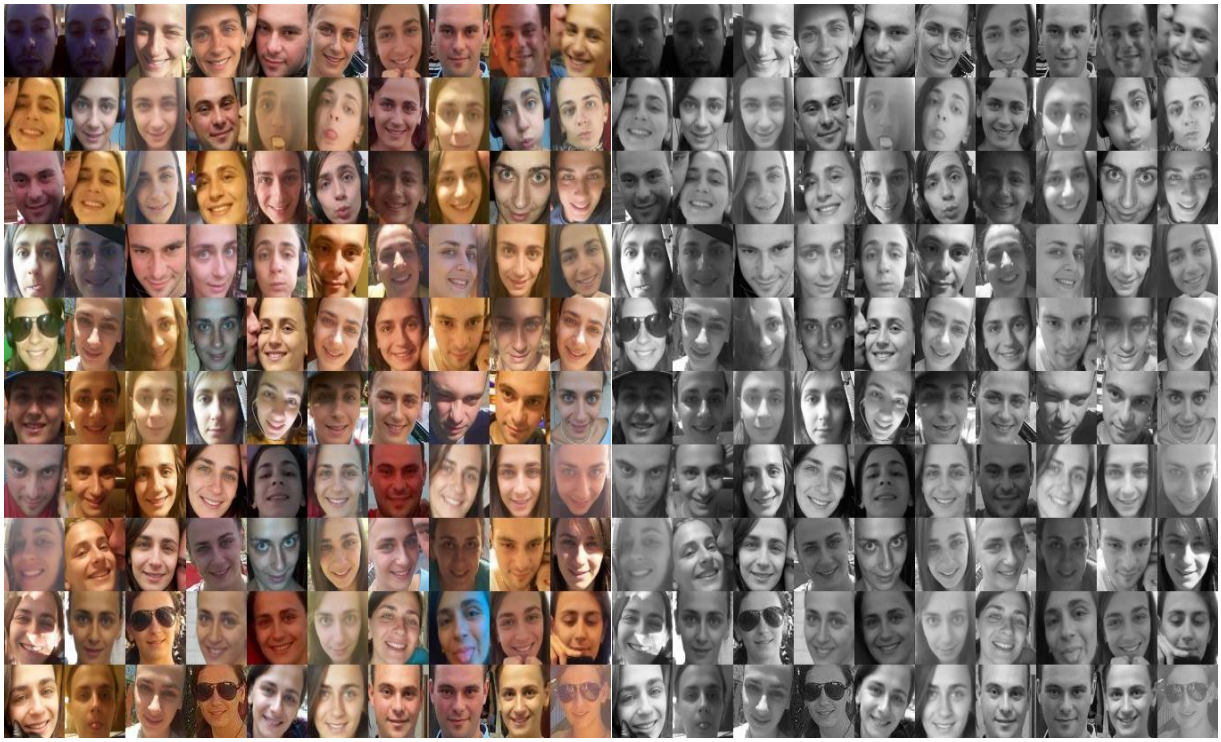


Фигура 14. Запазване на снимки със съответните имена

За ефективната работа на системата за разпознаване е необходимо да се събере възможно най-богата колекция от изображения, която да служат като база за тестовете. Важни е тези изображения да са с различни параметри и характеристики:

различен пол, различна форма на лицето, наличие на диоптрични очила, наличие на слънчеви непрозрачни очила, различни гримаси, наличие на шапка и различни прически на косата, различни ъгли на завъртане на главата.

Наличието на разнообразни изображения повишава възможностите на алгоритъма за разпознаване на лица, а една от съществените функционални възможности на предложената система е способността за засичане на лица с разнообразни характеристики. На фигури 15а) и 15б) са показани колажи от засечени във видео потока лица. При засичането изображенията се сканират и след това се изрязва и използва само частта, в която има лице. В паметта се пази снимката в цветен и черно-бял вариант, тъй като цветното изображение е нужно, за да се възстанови то при следващо показване на екрана в реално време, както и да присъства в базата данни, а черно-бялото е необходимо на алгоритмите както за засичане, така и за последващото разпознаване на лицето.



Фигура 15. Снимки на засечени от системата лица

а) цветни *б) черно-бели*

На фигура 16 са показани други варианти на засечени от системата лица, като сред тях присъстват такива с диоптрични очила, усмихнати гримаси и други.

Фигура 16. Снимки на засечени лица с различни характеристики

Функционални възможности за засичане и разпознаване на множесто лица

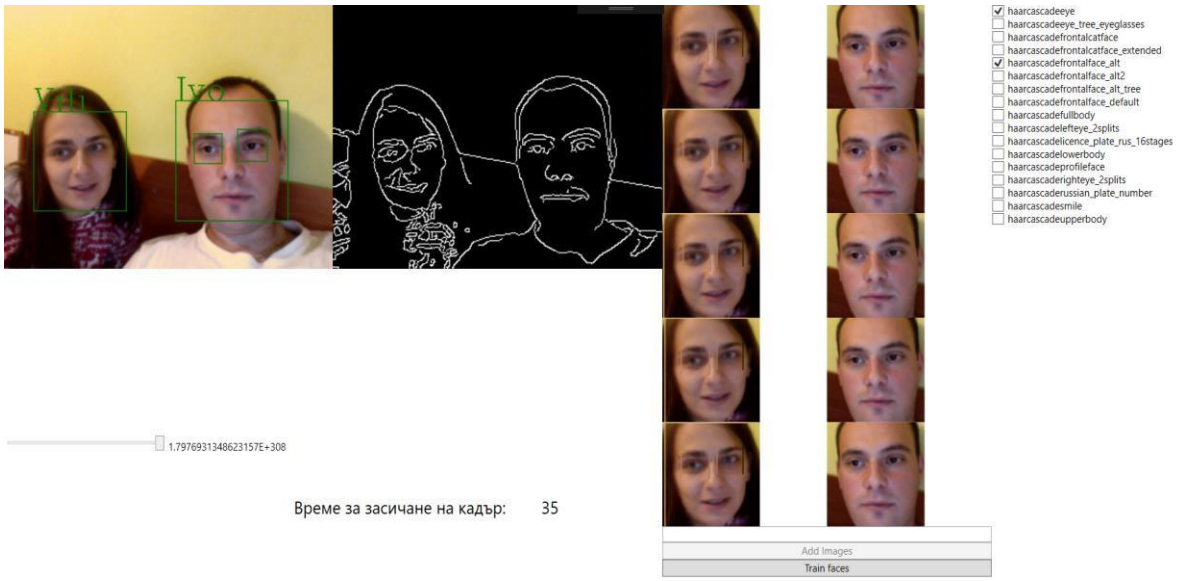
Един от основните тестове, през които трябва да премине подобна система, е следният: колко лица едновременно могат да бъдат засечени и разпознати в реално време. Възможността да бъде засено повече от едно лице е ключова за приложенията, в които може да бъде използвано.

Този тест е проведен чрез увеличавана на броя на хората, които попадат в кадъра на използваната камера, като изводите са следните: приложението успешно засича повече от едно лица едновременно. Същото е в сила и за разпознаването.

На фигура 17 е показана снимка на моментно действие на приложението. В случая са засечени и разпознати и двете лица, налични във видео потока.

В дадения случай са засечени и очите на едното лице (в дясно), докато засичането на очите на лицето вляво се вижда на отделните снети кадри вдясно от видео потока. Очите са разпознати, тъй като освен дескриптор за откриване на лице анфас, е избран и дескриптор за засичане на очи.

Лицата са разпзнати правилно на лицата, като имената са изписани над всеки правоъгълник, локализиращ лицето. На главния екран на приложението е изведено поле за *Време за засичане на кадър*, което в конкретния случай е 35 милисекунди за обработка. Засичането на лица се влияе в значително малка от наличната светлина.



Фигура 17. Засичане и разпознаване на две лица

При наличие на повече от две лица резултатите са напълно аналогични.

Функционални възможности за обслужване на множество различни клиенти

Клъстеризацията на системата предоставя възможност за обслужване на повече от един клиент.

При тестовете е симулирана връзка между много машини в клъстерна система. Ако в реални условия клъстерната система е инсталирана на множество машини тя ще работи по-бързо чрез получаване на информация (кадри) от няколко камери, обработването ѝ в реално време и връщане на резултати на всеки съответен свързан към системата клиент.

Тази функция придава изключителни възможности на приложението, а именно едновременното събиране на информация от множество източници и обработването ѝ в реално време. При наличие на повече от един източници, взимането на решения става в реално време и самият коефициентът на разпознаване е по-голям, тъй като наборът от данни, с които се обучава невронната мрежа е по-голям.

Тестове за време при засичане и разпознаване

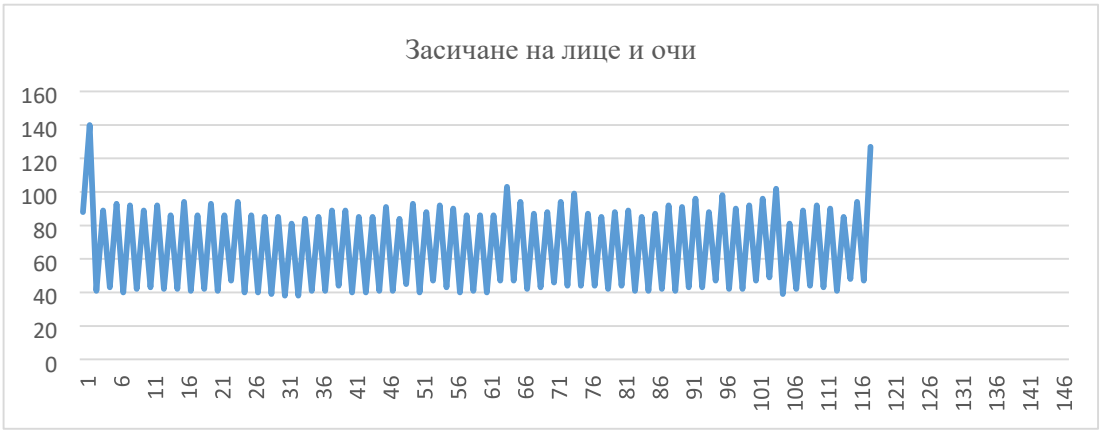
Тестовете за време се състоят в засичане наз времето в милисекунди съответно за засичане на едно лице, за разпознаване на едно лице и за засичане+разпознаване на налично лице.

Тестовете са направени с до 145 кадъра, като времето е измерено от системата в реално време.

Резултатите показват, че времето за засичане на лице е в диапазона 40-50 милисекунди (фиг. 18), а за засичане на лице и очи в рамките на лицето - 40-60 милисекунди (фиг. 19).

Фигура 18. Време за засичане на лице, като по абсцисната ос се отчитат брой кадри, а по ординатната – време в милисекунди

Получените резултати са задоволителни при характеристиките на камерата и машината, на която се извършват тестовете.



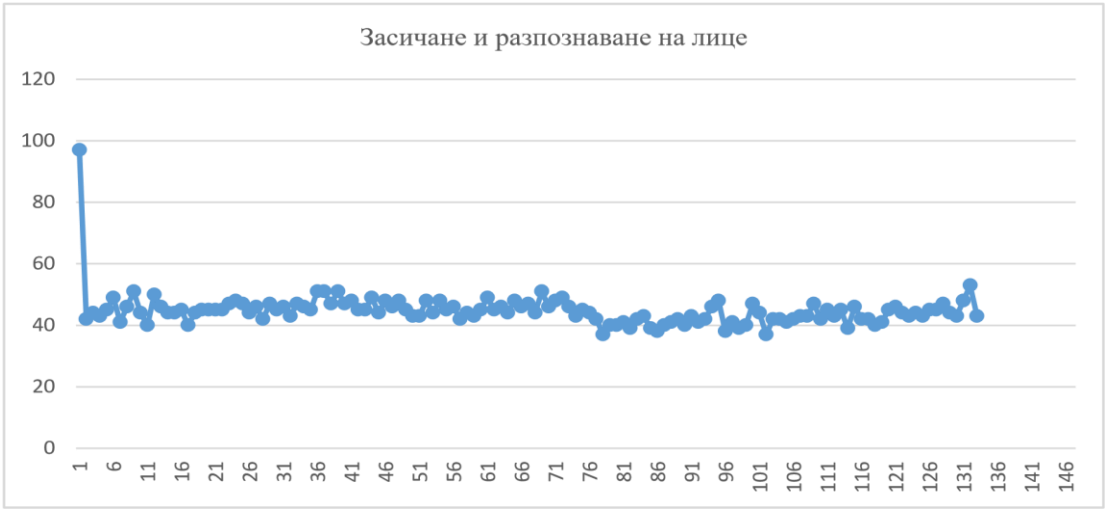
Фигура 19. Време за засичане на лице и очи, като по абсцисната ос се отчитат брой кадри, а по ординатната – време в милисекунди

Най-важен е фактът, че системата се вписва идеално във времевия диапазон за работа на система в реално време. След като приложението успява да засече лице, а дори и очи в рамките на лицето, за период по-малък от 100 милисекунди, може да се направи изводът, че системата ще засича лице, ако такова е налично, мигновено след стартирането ѝ, като човешкото око няма да отчете забавяне при засичането. При засичане и разпознаване на повече от едно лице системата има нужда няколко десетки милисекунди повече, което показва, че необходимото време не е право пропорционално на броя лица и функционалността за работа в реално време не е нарушена.

Другите два теста за време са за засичане+разпознаване на лице и засичане+разпознаване на лице и очи в рамките на лицето.

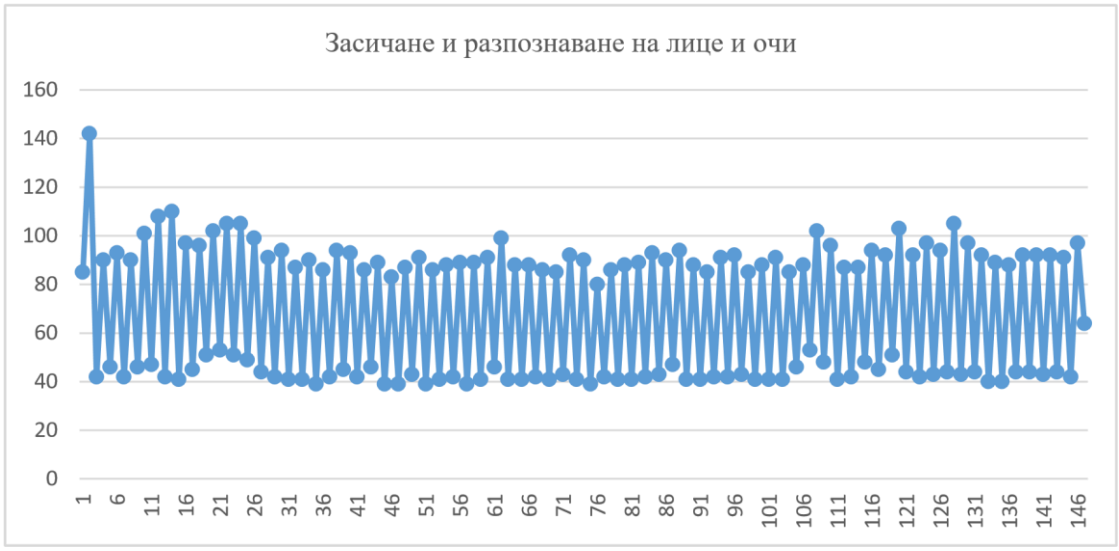
За засичане+разпознаване на лице са използвани 133 кадъра, а при засичането+разпознаване на лице и очи - 147 кадъра.

Процедурата по засичане+ разпознаване на едно лице от системата се изпълнява в рамките средно на 40-50 милисекунди (фиг. 20). Изводът е, че времето за засичане+ разпознаване не се различава значително от времето, необходимо само за засичанети на лицето, което е впечатляващ резултат за конкретните тестови условия.



Фигура 20. Време за засичане+разпознаване на лице, като по абсцисната ос се отчитат брой кадри, а по ординатната – време в милисекунди

Необходимото време за засичане+разпознаване на лице и очи заедно е малко поголямо от времетоза засичане+разпознаване само на лице и е в диапазона от 40-85 милисекунди, но почти се различава от този при тестовете само за засичане на лице и очи (фиг. 21).



Фигура 21. Време за засичане+разпознаване на лице и очи, като по абсцисната ос се отчитат брой кадри, а по ординатната – време в милисекунди

Резултатите като средно време от тези два теста са изцяло в допустимите рамки за засичане в реално време, като при други условия (пр. повече от едно лица в кадъра) времето ще бъде по-голямо, но със сигурност отново в допустимите граници.

Тестове при различно качество на видео потока

Освен времето за изпълнение на главните задачи за засичане и разпознаване, бързодействието на системата е тествано и при различно качество на получаваните кадри. Това показва устойчивостта на приложението при лошо качество на картината и е показател за минималните изисквания за надеждността му.

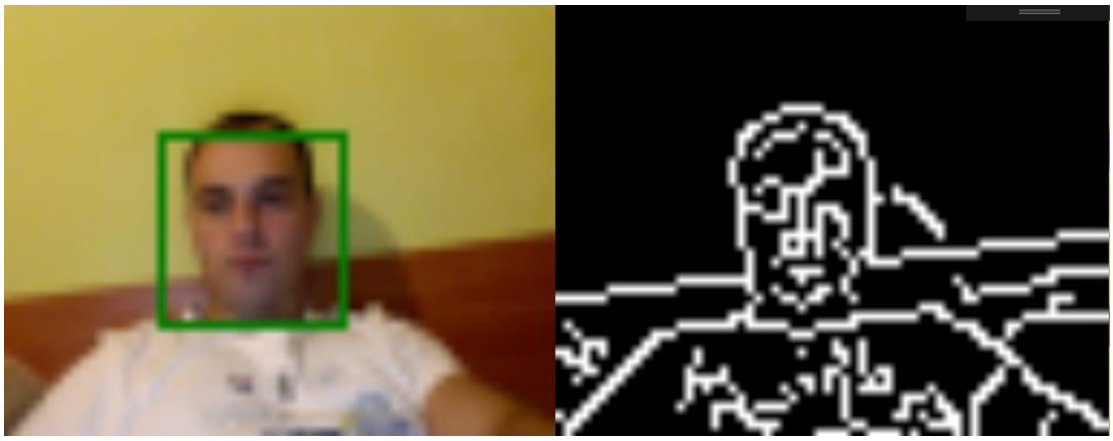
Направени са тестове с три различни вида резолюция с различни части от наличното изображение.

При изображение с резолюция 64x48 пиксела се достига време под 1 милисекунда. При следващия тест размерите на кадрите са 160x120 пиксела, като е запазена ¼ от целия кадър. В този случай времето за засичане се увеличава до 5-7 милисекунди и не се наблюдава разлика при разпознаването. При последния направен тест резолюцията е 320x240 пиксела. При този случай се наблюдават промени в бързодействието и на двата алгоритъма, като времето е 20 милисекунди за засичане и отново 20 милисекунди за разпознаване.

Основен извод от тази група тестове е, че не се наблюдава разлика във времето за засичане или разпознаване на лицето въпреки влошеното качество на изображението. Резултатите от проведените тестове са обобщени в Таблица 1. *Таблица 1. Резултати от тестове за време при различно качество на видео потока*

Резолюция	**Част от изображението**	**Време за засичане**	**Време за разпознаване**
64x48 px	1⁄10	<1ms	<1ms
160x120 px	1⁄4	5-7ms	5-7ms
320x240 px	1⁄2	20ms	20ms

Засичане на лице при влошеното качество на видео потока е илюстрирано на фигура 22.

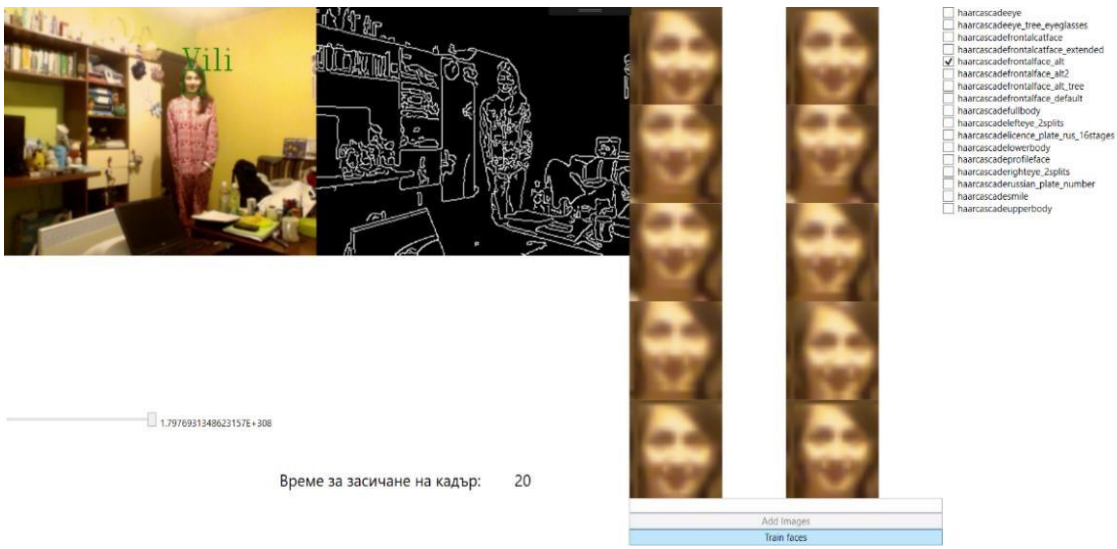


Фигура 22. Засичане на лице при ниско качество на кадрите

Тестове за отдалечено засичане и разпознаване

Системите за засичане и разпознаване на образи се използват най-вече в области, които са свързани с наблюдение на големи групи хора, които не във всички случаи се намират близко до камерата. По тази причина подобна система трябва да бъде устойчива и на разстоянието, на което се засича и разпознава лице.

След направени тестове за засичане и разпознаване при различни разстояния от сканиращата камера, изводите са, че текущата система засича и разпознава до 6 метра при влошено качество на видео потока и до 10 метра при реалната разделителна способност на тестовата камера от 640x480 пиксела. Засичане на отдалечен образ и изображенията на лицето, които са изрязани след процеса на засичане са показани на фигура 23. Качеството е влошено, но времето за засичане и разпознаване не се променя значително.



Фигура 23. Засичане и разпознаване на отдалечен образ

Коефициент на грешно засечени лица и незасечени присъстващи лица

В системите за засичане на образи има два основни критерия: *грешно засечени лица* и *незасечени, но присъстващи в кадъра лица*. При първия критерий алгоритъмът може да засече обект като лице в даден кадър, но това всъщност да не лице. Във втория случай обект, който представлява лице на даден човек, не бива засечено като такова от системата. И двата случая са нежелателни за една система за разпознаване и следователно са предизвикателство за преодоляване.

В текущата разработка са направени няколко оптимизации, в резултат на които се получава по-голям процент на точност при алгоритъма за засичане и оттам и при този за разпознаване на лица. В случая оптимизациите се насочени главно към проблема на грешното засичане, което спомага и за бързината на системата, защото грешно засечените области от даден кадър не се подават на невронната мрежа за разпознаване.

Главната оптимизация при търсенето на лице е свързана с намирането на очи в рамките на потенциално открито лице. Построено е следното правило: *наличие на лице* $\leftrightarrow$ *наличие на две несъвпадащи очи,* според което е възможно да има лице само там, където има две правилно позиционирани очи и обратно.

Първо се филтрират намерените правоъгълни области, в които очите не са на правилното разстояние едно от друго или излизат от рамките на дадения правоъгълник с лице в него по следния начин:

```
1 reference | Ivelin Kirilov Ivanov, 3 hours ago | 1 author, 1 change
public static List<Rectangle> FilterFacesRect(this List<Rectangle> faces, List<Rectangle> eyes)
{
    List<Rectangle> rect = new List<Rectangle>();

    foreach (var eye in eyes)
    {
        rect.AddRange(faces.Where(x => x.X <= eye.X && x.Y <= eye.Y).ToList());

    }
    return rect;
}
```

Филтърът, който служи за ограничаване на намерените очи, съдържа ограничителни проверки, свързани със самите разстояния между очите и това дали те се намират на приблизително една и съща координата по ординатната ос:

```
1 reference | Ivelin Kirilov Ivanov, 2 hours ago | 1 author, 1 change
public static List<Rectangle> FilterEyeRect(this List<Rectangle> eyes, List<Rectangle> faces)
{
    List<Rectangle> newEyesList = new List<Rectangle>();

    foreach (var face in faces)
    {
        var temp = eyes.Where(x => x.X > face.X && x.Y > face.Y && x.Right < face.Right && x.Bottom < face.Bottom)
            .OrderBy(x => x.Height)
            .ThenBy(x => x.Width)
            .ThenBy(x => x.Y)
            .ToList();
        if (temp.Count() > 1)
        {
            Rectangle[] leftRightEyes = new Rectangle[2];
            int minYdifference = Int32.MaxValue;
            int maxXdifference = Int32.MinValue;

            for (int i = 0; i < temp.Count; i++)
            {
                for (int j = 0; j < temp.Count; j++)
                {
                    if ((temp[i].Y - temp[j].Y) <= minYdifference
                        && (temp[i].X - temp[j].X) >= maxXdifference
                        && (temp[i].Right < temp[j].X || temp[i].X > temp[j].Right))
                    {
                        leftRightEyes[0] = temp[i];
                        leftRightEyes[1] = temp[j];
                    }
                }
            }
            if (leftRightEyes[0].X != leftRightEyes[1].X)
                return leftRightEyes.ToList();
        }
    }
    return newEyesList;
}
```

Въпреки че тези две филтриращи грешните лица проверки увеличават в известна степен грешките от другия тип (незасечени, но присъстващи лица, при които в дадения кадър присъства лице, което не може да бъде засечено от системата), те се извършват, защото без тях системата би давала по-голям процент грешни резултати. От друга страна грешка от вторият вид може да бъде избегната чрез увеличаване качеството на видео потока, като по този начин ясно ще се отличават детайли като очите и така системата ще работи с висока точност.

V. ЗАКЛЮЧЕНИЕ

В настоящата студия е представена реализирана софтуерна система за засичане и разпознаване на лица в реално време.

Системата обработва изпратеният в реално време от потребителя видео поток, като го разделя на кадри. Тези кадри се трансформират в черно-бели изображения. Следващият етап е подаване на текущите изображения на отделни възли на системата за обработване от невронна мрежа, реализираща алгоритъма ВиолаДжоунс. Накрая потвърдените за реални лица се подават на невронна мрежа, прилагаща принципа на анализ на главните компоненти, за да бъде установена самоличността на лицето. Преди допускане на ново изображение т, невронната мрежа бива обучена с последното състояние на базата данни от изображения за сравнение. След разпознаването, информацията се изпратеща обратно на потребителя. Процедурата по засичане и разпознаване може да бъде изпълнявана за множество различни източници на видео поток едновременно.

Софтуерът, който реализира системата, е написан на програмния език от високо ниво C#, като е използвана библиотеката с отворен код OpenCV.

Възможно бъдещи подобрение на предложената система е възможно с добавяне на хистограма на ориентираните градиенти - алгоритъм, подходящ за локализиране на хора в движение. Друга възможност е прилагане на алгоритъм за прихващане ръбовете на лицето и направата на 3D модел от намерените вектори. Тези подходи са подходящи за подобряване на ефективността на софтуерна система, работеща в реално време, зашото предлагат технология за бързо следене на лицето след засичането му и постигането на независимост на засичането и разпознаването от фактори като светлина, ъгъл, разстояние и други.

Библиография и използвани интернет източници

[1] Orts J., "Face Recognition Techniques," Madison, Wisconsin, 2014.

[2] West J., "Brief history of face recognition software," Face First, [Online]. Available: www.facefirst.com.

[3] Goldstein A., L. Harmon, A. Lesk, “Identification of human faces,” *Proceedings of the IEEE,* vol. 59, no. 5, pp. 748 - 760, 1971.

[4] Sirovich L., M. Kirby, “Low-dimensional procedure for the characterization of human faces,” *Journal of the Optical Society of America,* vol. 4, no. 3, p. 519–524, 1987.

[5] Pentland A., M. Turk, "Eigenfaces for recognition," Massachusetts Institute of Technology, Massachusetts, 1991.

[6] Mayhew S., "History of Biometrics," 2015. [Online]. Available: http://www.biometricupdate.com/201501/history-of-biometrics.

[7] Wechsler H., "Standarts Overview," in *Reliable Face, Recognition Methods, System Design, Implementation and Evaluation*, George Mason, Springer, 2007, pp. 95-98.

[8] ISO/IEC JTC 1/SC 29, “ISO/IEC 15444-1:2004”. USA Patent 800, 1 9 2004.

[9] Jolliffe T., Principal Component Analysis, second edition, Springer-Verlag, 2002.

[10] Laplante A., Real-time systems design and analisys, IEEE-Press, 2004.

[11] Juvva K., "Real-Time Systems," 1998. [Online]. Available: https://users.ece.cmu.edu/~koopman/des_s99/real_time/.

[12] Milo R., “BioNumbers,” Harvard University, 2010. [Online]. Available: http://www.bionumbers.hms.harvard.edu/.

[13] Rauber T., Parallel programming for Multicore and Cluster systems, Springer, 2013.

[14] Hausdorff F., "Basics of Set Theory, Berlin, 1914.

[15] Viola P., M. Jones, Robust Real-time Object Detection, Cambridge, 2001.

[16] Gao W., "On the doubt about margin explanation of boosting," *Science Direct,* vol. 203, no. 2, pp. 1-18, 2013.

[17] Krishan K., R. Mishra, "Principle Component Analys," in *Human and face detection and recognition*, Rourkela, 2011, pp. 9-12.

[18] Jauregui J., Principal component analysis with linear algebra, Philadelphia: Penn Arts & Sciences, 2012.