Универзитет у нишу

Електронски факултет

Катедра за рачунарство

**Праћење регуларности полагања тестова применом техника рачунарског вида**

- дипломски рад -

**Задатак:**

Упознати се са постојећим софтверским решењима која се користе за праћење регуларности онлајн полагања тестова. Идентификовати технике рачунарског вида које се могу искористити за реализацију оваквих система. У практичном делу имплементирати прототип система који употребом камере обезбеђује препознавање особе која полаже тест, врши детекцију нерегуларности у виду окретања и разговора, те генерише извештај са видео доказом спорних ситуација.

**Mентор:** проф. др Александар Милосављевић **Кандидат:** Ивана Миливојевић 16704

Комисија:

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум пријаве: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум предаје: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум одбране: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ниш, 2022.

**Садржај**

[**Увод** 3](#_Toc119426319)

[**Софтвери за праћење регуларности полагања тестова** 4](#_Toc119426320)

[Аутоматско надгледање онлајн тестирања 4](#_Toc119426321)

[**Технике рачунарског вида за праћење регуларности полагања тестова** 7](#_Toc119426322)

[Детекција објеката 7](#_Toc119426323)

[**Viola-Jones (Haar) детектор** 8](#_Toc119426324)

[**HOG детектор** 10](#_Toc119426325)

[**R-CNN детектор** 13](#_Toc119426326)

[**YOLO детектор** 16](#_Toc119426327)

[**SSD детектор** 17](#_Toc119426328)

[Детекција и препознавање лица 19](#_Toc119426329)

[**Eigenfaces** 20](#_Toc119426330)

[**Fisherfaces** 21](#_Toc119426331)

[**LBP** 22](#_Toc119426332)

[**Дубоко учење** 23](#_Toc119426333)

[Одређивање положаја главе 24](#_Toc119426334)

[Праћење покрета очију 26](#_Toc119426335)

[Детекција говора 28](#_Toc119426336)

[**Имплементација система за праћење регуларности полагања тестова** 29](#_Toc119426337)

[Компоненте система 31](#_Toc119426338)

[**Закључак** 40](#_Toc119426339)

[**Литература** 41](#_Toc119426340)

# **Увод**

Пандемија настала услед ширења вируса Ковид-19 утицала је на све аспекте живота људи, укључујући образовање. У условима које је проузроковала ова ситуација било је неопходно прилагодити начин извођења наставе како би се школска година одвијала несметано. Многе образовне институције су морале да пређу на хибридни или у потпуности онлајн модел одржавања наставе, те је један од највећих изазова био како пронаћи адекватан начин за оцењивање ученика.

Онлајн тестирање је већ било у великој мери заступљено у пракси зато што олакшава и убрзава процес креирања тестова, полагања, као и анализу резултата. Флексибилност и могућност рада од куће разлози су због којих је велики број наставника одабрао овакве тестове као меру оцене. Наравно, тиме се отвара простор за различите начине преписивања. Неки од начина да се преписивање спречи су ограничење да сви морају полагати тест истовремено, генерисање већег броја различитих питања, приказ питања и понуђених одговора у различитом редоследу, увођење временског ограничења за одговор за свако питање, онемогућавање враћања на претходно питање итд. Међутим, ове мере некада нису довољне јер ученици могу да пронађу начин да комуницирају са неким или потраже одговоре у литератури и на интернету. Из тог разлога направљени су софтвери за надгледање полагања онлајн тестирања. У наставку ће се учеником ословљавати особа која полаже тест, али ови софтвери, поред примене у образовним институцијама, имају примену и у различитим компанијама и организацијама за потребе тестирања током интервјуа за посао, семинара, добијања сертификата...

Најчешће се прате микрофон, камера и екран ученика. Једно од решења је да дежурна особа надгледа ученика током полагања у реалном времену. Друго решење подразумева снимање ученика током полагања и касније прегледање (убрзане верзије) тог снимка ради контроле регуларности. Предност оваквог решења је што ученик и дежурни не морају у заказано време бити присутни, али у овом случају нема могућности да дежурна особа реагује у реалном времену на неправилности због којих је можда требало санкционисати ученика или онемогућити му наставак полагања теста. Треће решење је аутоматско нагледање полагања коришћењем алгоритама вештачке интелигенције које у великој мери мења особу која би била задужена да лично, све време, надгледа ученике. Систем генерише упозорења током полагања (која могу алармирати надлежну особу да се прикључи полагању) и бележи све нерегуларне ситуације које су детектоване и које се касније могу додатно прегледати како би се потврдиле неправилности. [1]

У овом раду биће обрађене методе рачунарског вида које се могу користити за аутоматску обраду података са камере приликом полагања онлајн тестова. Рад је организован у пет поглавља. Након уводног дела, у другом поглављу је дат преглед постојећих софтверских решења за проблем надгледања онлајн тестова. У трећем поглављу прво су обрађене технике и алгоритми рачунарског вида који се могу применити за детекцију објеката, на које се надовезују алгоритми за детекцију и препознавање лица. Затим су описани начини за одређивање положаја главе, праћење покрета очију и детекцију причања на видео снимку. У наредном поглављу описана је конкретна имплементација једноставног прототипа система који се може користити за надгледање регуларности полагања онлајн тестова обрађивањем фрејмова са камере. Систем детектује нерегуларне ситуације (у виду одсуства особе у кадру или присуства више особа, коришћења мобилног телефона, окретања и гледања са стране, разговора), врши препознавање лица, приказује упозорења ученику за време теста и све детектоване нерегуларности бележи у видео фајл. Закључно поглавље осврће се на претходна поглавља и сумира теоријске и практичне аспекте описаног проблема.

# **Софтвери за праћење регуларности полагања тестова**

Постоје различити софтвери који су развијени у циљу омогућавања реализације учења на даљину. Неки од њих укључују могућност тестирања, а има и посебних софтвера само за ову намену. Углавном се ради о веб апликацијама, док је неке апликације потребно инсталирати на свом уређају. За креирање кратких тестова који не захтевају висок степен сигурности и провере, најједноставније је користити неки од веб сајтова за генерисање квизова и тестова који немају никакав вид контроле.

Када су у питању озбиљнији тестови који захтевају висок степен сигурности, треба одабрати (у зависности од потреба и буџета организације) неки од напреднијих софтвера за тестирање који нуде и надгледање регуларности полагања. Као најважнији захтеви које је потребно да напреднији софтвери за онлајн тестирање испуне могу се издвојити: једноставност коришћења, превенција варања, сигурност, скалабилност, прилагодљивост и интеграција са другим сервисима. Једноставност коришћења је веома важна јер софтвер треба да корисницима пружи добро корисничко искуство. Технике за детекцију нерегуларности су неопходне како би се очувао интегритет теста. Потребно је обезбедити сигурност осетљивих података као што су сами тестови и лични подаци корисника. Скалабилност има важну улогу када је у питању употребљивост система зато што може бити потребно да више хиљада корисника истовремено полаже тестове. Како корисници апликације могу бити различите институције и организације, имаће различите потребе, те треба обезбедити могућност избора функционалности које су захтеване за реализацију одређеног тестирања. Још једна важна особина је да систем треба да буде једноставан за интеграцију са постојећим платформама за учење. [2]

Када је реч о надгледању полагања онлајн тестирања, у уводном поглављу је напоменуто да једна група ових софтвера врши аутоматско надгледање и не захтева константно ангажовање дежурне особе током тестирања. Овакви софтвери се све више користе у пракси. У наставку ће бити дат преглед функционалности софтвера за аутоматско праћење тестирања и неколико примера популарних производа који су доступни и користе их многобројне институције и организације.

## **Аутоматско надгледање онлајн тестирања**

Први корак у процесу полагања тестова углавном је пријављивање корисника које може бити имплементирано на више начина, али најчешће се користи и-мејл (енг. email) пријава. Пре самог покретања теста може се од ученика затражити да коришћењем камере покаже своју идентификациону картицу или омогући систему да забележи његову слику и упореди је са сликом из базе података. У оквиру апликације може постојати више модула који су задужени за обраду различитих података. Модул за праћење аудио података захтева да ученик има повезан микрофон и да обезбеди тишину у просторији у којој ће радити тест. Уколико тест захтева гласовну активност ученика може се испитивати да ли је учеников глас исти током трајања теста и да ли се поклапа са гласом тог ученика из базе података, као и да ли ученик разговара са неким. Надгледање звука може спречити да неко други ради тест уместо ученика и да му нека особа или апликација помажу „добацивајући“ му одговоре на питања. Модул који је задужен за праћење екрана ученика углавном санкционише сликање и снимање прозора у ком је отворен тест, напуштање теста и отварање других страница, апликација или фајлова. Модул за обраду података са камере захтева да камера ученика буде укључена током полагања теста и може да контролише да ли у кадру има других особа, недозвољених предмета попут мобилних телефона, књига и папира, да врши препознавање лица, могу се пратити покрети ученика и детектовати када он гледа са стране. Софтвер за надгледање полагања додатно може вршити и онлајн претрагу како би проверио да ли је садржај теста „процурео“. Поједини софтвери су у могућности да уз помоћ вештачке интелигенције лоцирају и уклоне садржај теста који је неауторизовано доспео на интернет. [3]

Једно од водећих решења је *Honorlock*, први сервис за надгледање онлајн тестирања који комбинује аутоматско надгледање са људским дежурањем. Тестирање надгледа вештачка интелигенција и ако детектује нерегуларности алармира дежурну особу да се прикључи сесији у реалном времену. Предуслови за израду теста су да особа која полаже тест буде сама у просторији, да буде тишина, да рачунар на коме се ради тест има само један монитор и да корисник поседује 360° камеру како би се скенирала просторија у којој се налази. *Honorlock* користи софтвер за закључавање веб претраживача који онемогућава приступ другим веб сајтовима, при чему закључава и одређене пречице на тaстатури, онемогућава минимизирање претраживача и напуштање теста пре предаје. Наставници имају могућност да по жељи специфицирају којим веб сајтовима желе да омогуће приступ ученицима током тестирања. Могућа је директна интеграција са платформама за учење. Такође, софтвер идентификује „процурели“ садржај теста на интернету и предузима кораке за његово уклањање. Још једна функционалност која је уведена је да систем детектује ако ученик покуша да приступи материјалу за учење током теста путем неког другог уређаја и бележи снимак екрана током трајања приступа. Овај систем не користи биометријске методе за идентификацију ученика попут препознавања лица, већ пре почетка теста услика ученика који држи своју идентификациону картицу и након 60 секунди омогућава полагање теста. [4]

*ProctorEdu* је још једно решење за онлајн надгледање, снимање и евалуацију корисничког понашања током онлајн тестирања, без потребе преузимања било каквог софтвера. Функционалност надгледања је интегрисана са платформама за учење и омогућава контролу удаљеног тестирања уживо или у аутоматском моду. Софтвер нуди биометријску аутентификацију, аутоматски опоравак након губљења конекције, могућност повезивања додатне камере са мобилног телефона за преглед простора где се налази ученик у 360° (камера се повезује скенирањем QR кода). Систем дежурну особу обавештава о нерегуларним активностима у реалном времену и омогућава да ученици комуницирају са њом путем чета, видеа или аудија уколико буду имали неких питања. Обезбеђена је функционалност препознавања лица током трајања теста, детекција буке, као и контролисања покушаја претраге на интернету. Извештај о нерегуларностима се генерише у видео и пдф формату. [5]

*Mercer* ***|*** *Mettl* такође нуди безбедно тестирање. Овај софтвер користи више од 150 универзитета широм света и обављено је више од 12 милиона тестова ове године. Нуди избор између надгледања уживо и аутоматског надгледања. Може се користити као комплетна платформа за тестирање, а и као сервис за удаљено надгледање тестирања којим се проширује нека друга платформа за тестирање. Систем врши видео и аудио надгледање. Детектује уколико кандидат није присутан или је присутна особа која није она којом се представља, присуство мобилних телефона и разговор са другим особама. Током трајања теста се у одређеним временским интервалима скенира слика ученика и упоређује са његовом сликом из базе података. Тестови се могу полагати и на уређајима попут мобилног телефона и таблета, али за тестове који захтевају већу сигурност препоручљиво је полагати их на лаптоп или десктоп рачунарима. У понуди је и *Mettl Secure Browser* који искључује све екстерне портове, спречавајући ученика да повеже секундарни екран. Browser такође искључује све софтвере за дељење садржаја и веб сајтове и не допушта напуштање прозора у коме се ради тест пре предаје теста. AI модул је обучен да детектује до 18 типова нерегуларности. [6]

Examus је један од најфлексибилнијих сервиса за онлајн надгледање. Користи веб камеру за препознавање лица и детекцију емоција, праћење погледа и детекцију других особа. Нуди два главна производа: Proctor AI – софтвер за онлајн надгледање коришћењем вештачке интелигенције и Examus EQ – алат који надгледа пажњу и ангажовање особа током похађања онлајн курсева. Proctor AI комбинује податке из три извора (видео, аудио и радна површина рачунара) и онда их анализира коришћењем алгоритама базираних на неуронским мрежама како би креирао детаљан извештај о корисниковом понашању. Образовне институције, као клијенти, добијају тај извештај праћен видео доказом. У извештају се налазе линкови који нас одводе директно до релевантног дела видео доказа где можемо лично проверити шта се дешавало у том моменту. Бележе се ситуације попут гледања ван екрана, причања, комуницирања са другим особама, промена прозора на екрану, покушајa дељења екрана или звука. Постоји могућност укључивања дежурне особе која ће у реалном времену проверити забележене нерегуларности. [7]

# **Технике рачунарског вида за праћење регуларности полагања тестова**

Рачунарски вид је поље рачунарске науке које ради на томе да омогући рачунарима да виде, идентификују и обрађују дигиталне слике на сличан начин као што то чини људски вид. Тежи се разумевању и аутоматизацији задатака које визуелни систем човека може да уради. Као научна дисциплина, рачунарски вид се бави теоријом вештачких система која издваја информације из слика. Као технолошка дисциплина, настоји да примени теорије и моделе за изградњу система рачунарског вида. [8]

Када је реч о апликацијама за праћење полагања онлајн тестова, технике рачунарског вида можемо применити у оквиру модула који обрађује податке са камере. Неке од техника које се могу применити су детекција објеката, детекција и препознавање лица, одређивање положаја главе и праћење покрета очију и усана особе која полаже тест.

## **Детекција објеката**

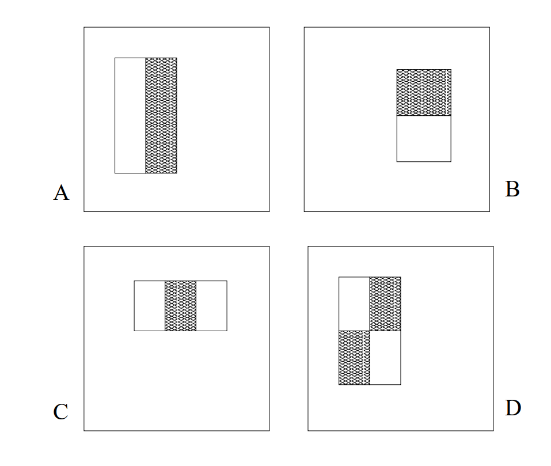
Детекција објеката је рачунарска технологија која спада у област рачунарског вида и бави се проналажењем објеката који припадају одређеним класама (нпр. људи, зграде, аутомобили) на дигиталним сликама и видео снимцима. [9] Свака класа објеката има специфичне особине, такозване фичере (енг. features), који одређују припадност објекта тој класи. Детекција објеката је један од основних проблема у рачунарском виду и има широку примену у свакодневном животу (нпр. за детекцију пешака и аутомобила, препознавање регистарских таблица, анализу слика и снимака у спорту и медицини). Даје нам информацију о томе који објекат је на слици и где се он налази.

Методе за детекцију објеката се генерално могу поделити на оне које се заснивају на традиционалним техникама обраде слика и оне које су базиране на дубоком учењу. Код метода које су засноване на традиционалним техникама издвајају се фичери објеката коришћењем неког од детектора фичера и након тога користити се нека од техника за класификацију. Најпознатији примери оваквих детектора су Viola–Jones (Haar) и хистограм оријентисанх градијената (HOG, енг. Histogram of Оriented Gradients). Технике базиране на дубоком учењу најчешће се ослањају на конволуционе неуронске мреже (CNN, енг. Convolutional Neural Networks). Поступак детекције се може реализовати кроз две фазе, а може и у оквиру једне. Детектори који раде у 2 фазе обављају два задатка: први је проналажење произвољног броја региона објеката, а други је класификација сваког од њих и одређивање правоугаоника којим се објекат може уоквирити (*bounding box* регресија). Ови детектори имају велику поузданост, али су спорији од једнофазних. Најпознатији двофазни детектори су R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN и G-RCNN. Једнофазни детектори предвиђају оквирне правоугаонике на слици без претходно издвојених региона од интереса третирајући детекцију објеката као регресиони проблем. Бржи су, структурно једноставнији и могу се користити код апликација од којих се очекује да раде у реалном времену, али су и нешто мање поуздани од једнофазних. Најпознатији детектори из ове групе су YOLO (енг. You Only Look Once), који има више верзија, SSD (енг. Single Shot MultiBox Detector) и RetinaNet. Раније методе за детекцију могу бити ограничене услед постојања комплексне позадине, делимично прекривених објеката, лошег осветљења и шума, док су технике дубоког учења значајно отпорније на наведене проблеме. Данас су рачунари много ефикаснији него што је то било раније и тежи се паралелизацији процеса, тако да дубоке конволуционе неуронске мреже у комбинацији са убрзањем које доноси графички процесор дају веома добре резултате и омогућавају детекцију објеката у приближно реалном времену. Недостатком ових метода може се сматрати потреба за великим бројем тренинг слика које треба ручно означавати. [10]

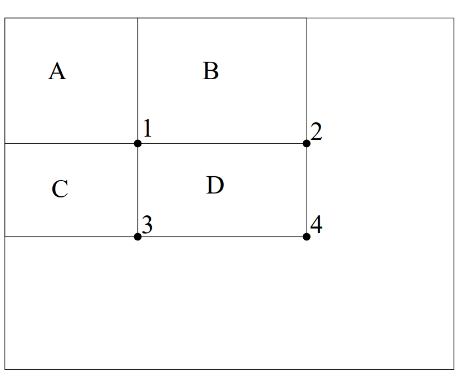
У наставку ће бити дат преглед неколико детектора објеката који су били значајни за развој ове области, почевши од Viola-Jones детектора који се први појавио, преко HOG детектора који се појавио пар година касније, до детектора који користе неуронске мреже као што су различите варијанте R-CNN-a, YOLO и SSD.

### **Viola-Jones (Haar) детектор**

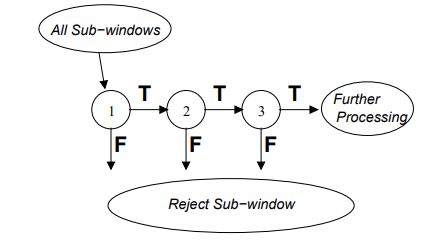
Paul Viola и Michael Jones су у свом раду [11] из 2001. године представили метод за детекцију објеката у реалном времену реализован кроз примену неколико нових техника. Првобитно је био намењен аутоматској детекцији лица, али може се користити за детекцију било ког типа објеката.

За детекцију је коришћен скуп фичера који подсећају на Haar-ове таласе (одакле и потиче назив детектора), приказан на слици 1. Дефинисане су три врсте фичера: фичери са два (A и B) суседна правоугаоника исте величине која су хоризонтално или вертикално поравната, три (C) и четири (D) суседна правоугаоника. Вредност фичера се рачуна тако што се сума пиксела у белим регионима одузима од суме пиксела у сивим. Прва два фичера на слици (A и B) се користе за детекцију ивица, трећи (C) за детекцију линија, а четврти (D) за детекцију косих линија.

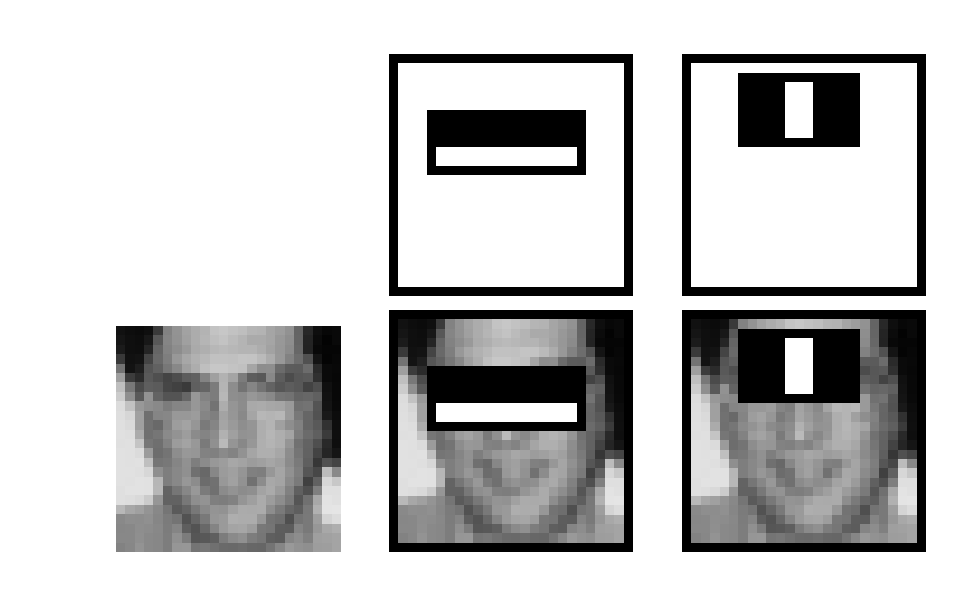
Слика 1: Haar фичери [11]

Аутори су приказали нови начин за репрезентацију слике назван „интегрална слика“, који омогућава да се фичери који се користе при детекцији брзо израчунају. Интегрална слика може бити израчуната једним проласком кроз оригиналну слику и може се посматрати као матрица истих димензија као оригинална слика, с тим што на локацији (x, y) садржи суму пиксела са оригиналне слике изнад и лево од тачке (x, y). Омогућава израчунавање суме пиксела произвољног правоугаоног региона коришћењем вредности у само 4 тачке. На слици 2 се може видети да сума пиксела у оквиру правоугаоника D може бити израчуната на основу вредности интегралне слике у тачкама 1, 2, 3 и 4. Вредност интегралне слике у тачки 1 је сума пиксела у правоугаонику . Вредност у тачки 2 је , у тачки 3 је , а у тачки 4 је . Сума пиксела у оквиру правоугаоника D би била , односно, . За два правоугаоника сума може бити израчуната на основу вредности у 6 тачака, за случај 3 правоугаоника на основу 8, а за случај 4 правоугаоника на основу 9 тачака. Када се интегрална слика једном израчуна, фичери било које величине на било којој локацији могу се израчунати за константно време.

Слика 2: Рачунање суме пиксела у оквиру правоугаоног региона [11]

За прозор димензија 24×24 се израчунавају фичери (око 180 000 фичера различитих величина и позиција). Иако се појединачни фичери израчунавају прилично једноставно и брзо, рачунање комплетног скупа фичера је захтевно и скупо. Да би се издвојио мањи скуп најважнијих фичера из великог скупа фичера и тренирао класификатор коришћен је алгоритам заснован на *AdaBoost* техници за учење. Свака фаза *boosting* процеса даје нови слаби класификатор који зависи само од једног фичера (бира се фичер који има најмању грешку класификације). Овиме се број фичера смањује на око 6000. Слаби класификатори не могу самостално да класификују слику, али њихова линеарна комбинација може представљати јак (добар) класификатор. За детекцију објеката је коришћена каскада јаких класификатора који се примењују један за другим, при чему је сваки класификатор у каскади комплекснији од претходног. На слици 3 је илустрован процес класификације. Ако прозор не прође први класификатор (не садржи објекат), бива одбачен и не процесира се даље. Ако прође, пропушта се кроз наредни класификатор и процес се понавља. Прозор који прође све класификаторе сматра се прозором који садржи објекат. Циљ је брзо одбацити регионе без објеката и даље процесирање вршити само над регионима који су обећавајући, односно који потенцијално садрже објекат.

Слика 3: Каскада класификатора [11]

У оригиналном раду је детекција објеката демонстрирана на примеру детекције лица. На слици 4 приказана су прва два фичера селектована *AdaBoost* алгоритмом која могу чинити први класификатор у каскади. Први фичер наглашава особину лица да је регион око очију углавном тамнији од региона горњег дела образа, док други фичер осликава особину да је регион очију тамнији од региона носа. Сваки од класификатора у каскади од 38 слојева трениран је коришћењем AdaBoost процедуре фронталним сликама лица које су скалиране на резолуцију 24×24 (како би се поклопиле са прозором) и сликама на којима нема лица са којих су издвајани потпрозори 24×24. Број фичера у првих 5 слојева је 1, 10, 25, 25 и 50, респективно. У процесу детекције детектор обилази слику која се тестира методом клизајућег прозора у више величина, при чему се детектор (прозор) скалира, а не слика, како би се искористиле особине интегралне слике. С обзиром на то да ће бити генерисано више детекција за исто лице, потребно је на крају те детекције свести на једну. Све детекције се раздвајају у непреклапајуће подскупове, при чему ће се две детекције наћи у истом подскупу ако им се оквирни региони преклапају. Сваки подскуп даће једну коначну детекцију чије границе су просек граница свих детекција у том подскупу. [11]

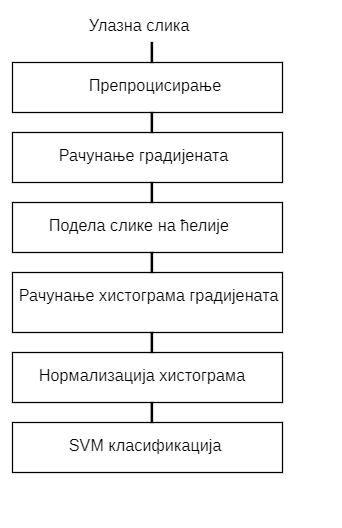
Слика 4: Прва два фичера селектована AdaBoost техником [11]

Овај алгоритам ради са сивим (енг. grayscale) сликама, детектује објекте на сликама независно од њихове локације и величине и инваријантан је на промене у осветљењу. Једна од највећих предности овог детектора у односу на остале је велика брзина детекције, а недостатак што је склон лажно позитивним детекцијама и детектује углавном само фронтално постављене објекте. Иако је објављен пре доста година, и даље се често користи првенствено због своје брзине извршавања.

### **HOG детектор**

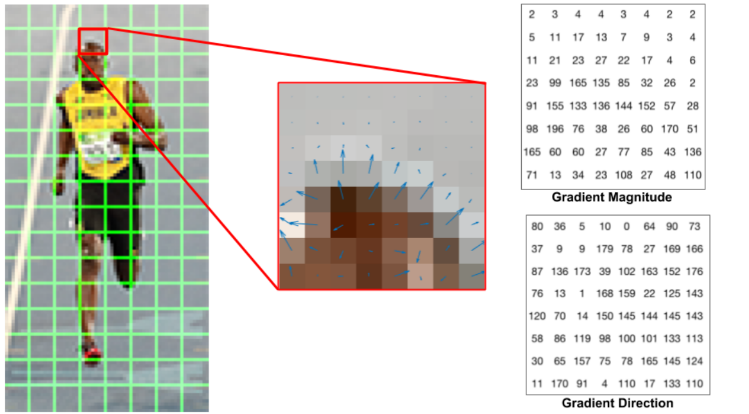
У раду [12] из 2005. године показано је да се за детекцију објеката на слици могу успешно користити HOG фичер дескриптор слике и линеарни SVM класификатор. Аутори су се фокусирали на детекцију пешака на слици, али детектор се може применити и за детекцију објеката других класа.

Фичер дескриптор је репрезентација слике којом се издвајају корисне информације из ње, а одбацују ирелевантне. Код HOG фичер дескриптора се улазна слика димензија 64×128×3 (канала) конвертује у вектор фичера дужине 3780, а као фичери користе се хистограми смерова оријентисаних градијената. Градијенти слике (x и y изводи) су корисни зато што је њихова магнитуда велика око ивица и ћошкова (региони наглих промена интензитета), а они носе много више информација о облику објекта него равни региони. На слици 5 приказан је процес детекције објеката HOGдетектором. Први корак је препроцесирање слике. У оригиналном раду HOG дескриптор фичера је рачунат над регионима слике величине 64×128, али у општем случају слика може бити било које величине. Посматрају се региони слике са различитим скалирањем и на различитим локацијама. Једино ограничење је да регион који се обрађује има фиксни *aspect ratio* (у нашем случају 1:2).

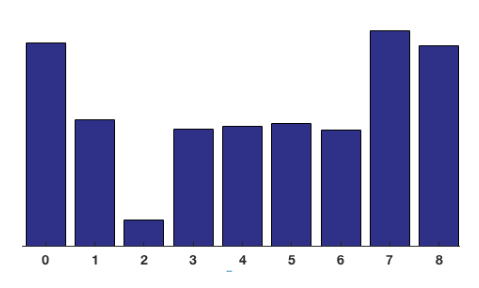
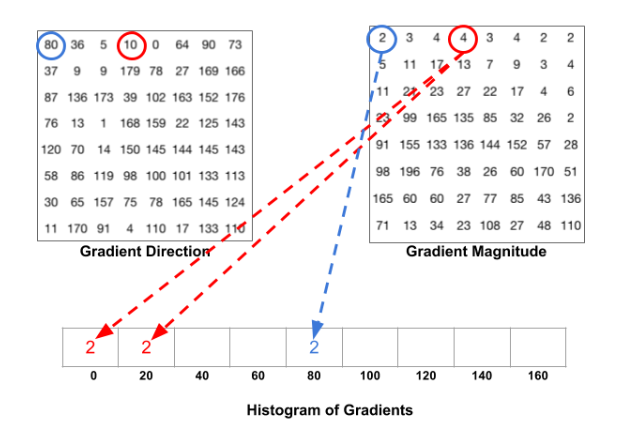
Након тога је потребно израчунати хоризонталне и вертикалне градијенте, што се може постићи филтрирањем слике Собел филтером са кернелом величине 1. Магнитуда и угао се рачунају по следећим формулама: , , где су и градијенти по x и y осама. На слици 6 лево приказана је слика особе која трчи, а десно можемо приметити да x-градијент издваја вертикалне линије, а y-градијент хоризонталне. Магнитуда градијената се јавља где год постоје нагле промене у интензитету, тако да финална слика садржи истакнуте све ивице. Особа је у првом плану, а већина ирелевантних информација је уклоњена. У сваком пикселу градијент има магнитуду и смер. Код слика у боји рачунају се градијенти за сва три канала, па се за магнитуду у сваком пикселу узима максимална магнитуда та три градијента, а за угао онај угао који одговара максималној магнитуди.

Слика 6: Део улазне слике, његови x и y градијенти и магнитуда градијената, респективно [13]

Слика 5: Процес детекције HOG детектором

Затим се слика дели на ћелије од 8×8 пиксела и хистограм градијената се рачуна за сваку ћелију. Део слике 8×8 садржи вредности пиксела. Градијент овог дела садржи 2 вредности (магнитуду и смер) за сваки пиксел, што даје вредности. Индивидуални градијенти пиксела могу садржати шум, али хистограм над регионом од 8×8 пиксела је много отпорнији на шум. Ове димензије слике и ћелија су одабране за потребе детекције пешака и биле су довољно велике да се издвоје значајни фичери. На слици 7 приказана је подела слике на ћелије (лево). У средини је издвојена једна ћелија чији су смерови градијената илустровани смером стрелица, а магнитуде дужинама стрелица. Смерови стрелица показују смер промене интензитета, а њихова дужина колико је велика разлика. Десно су бројевима представљене вредности магнитуда и смерова градијената за пикселе издвојене ћелије. Углови су између 0° и 180° уместо 0° и 360° зато што су посматрани „неозначени“ градијенти код којих су градијент и његова негативна варијанта представљени истим бројем. Y - оса представља 0°.

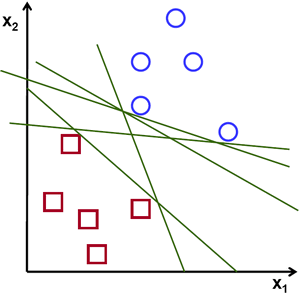
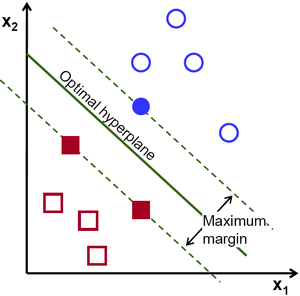
Слика 7: 8×8 ћелије (лево), градијенти ћелије представљеним стрелицама (средина), градијенти ћелије представљени бројевима [13]

Следећи корак је креирање хистограма градијената у оквиру ових 8×8 ћелија, илустровано на слици 8. Хистограм је низ од 9 елеменета који одговарају угловима 0°, 20°, 40°, …, 160°. Елемент низа је одабран на основу смера, а вредност елемента на основу магнитуде градијента. Градијент означен плавом бојом има угао (смер) од 80° и магнутуду 2, тако да се вредност 2 додаје елементу на 5. позицији. Градијент означен црвеном бојом има угао од 10° који је на средини између 0° и 20°, тако да се његова вредност 4 додаје 1. и 2. елементу (по 2). еАко је угао већи од 160°, он је између 160° и 180°, а како су углови од 0° и 180° еквивалентни, његова вредност би се пропорционално уписала у елементе намењене угловима од 0° и 160°. У десном делу слике 8 је приказано како би изгледао хистограм за одабрану ћелију са слике 7. Како у хистограму доминирају вредности близу 0° тј. 180°, може се закључити да ова ћелија вероватно садржи ивицу јер су градијенти углавном усмерени на горе или доле.

Слика 8: Рачунање хистограма градијената (лево), график хистограма (десно) [13]

Како би дескриптор био инваријантан на промене у осветљењу, треба нормализовати хистограм. Нормализација подразумева дељење сваке вредности вектора са дужином вектора. Посматрамо блок 16×16, он има 4 хистограма који надовезивањем формирају вектор од 36 елемената који се нормализuje. Прозор се онда помера за 8 пиксела и процес се понавља. Да би се израчунао коначни вектор фичера за комплетно парче слике, вектори дужине 36 се конкатенирају и креирају један велики вектор од 3780 елемената (, где је број позиција на којима може да се нађе прозор 16×16). Овај вектор се сада користи за тренирање SVМ класификатора. Примењују се пирамида слика и метода клизајућег прозора како би се детектовали објекти различитих величина. [13]

SVM или машина са векторима подршке је техника класификације код које је циљ пронаћи оптималну хиперраван (границу одлуке) у вишедимензионалном простору која раздваја податке који припадају различитим класама. Димензионалност простора је одређена бројем фичера. Уколико није могуће у потпуности поделити податке, треба наћи хиперраван која максимизује маргину, односно минимизује грешке у класификацији. Маргина представља најкраће растојање од хиперравни до најближег тренинг податка који припада било којој класи. На слици 9 је лево приказано више могућих хиперравни, а десно оптимална хиперраван у 2D простору. Очекује се да хиперраван са већом маргином буде прецизнија приликом класификације непознатих података у односу на хиперраван са мањом маргином. [14] Тренинг подаци који се налазе на самим маргинама се називају вектори подршке (енг. support vectors) и они се најтеже класификују, али дају највише података о самој класификацији. За случај тренинг скупа података који се не може линеарно раздвојити, проблем треба свести на проблем линеарног раздвајања, што се постиже нелинеарном трансформацијом тренинг скупа података у вишедимензионални простор где је могуће извршити линеарну поделу података.

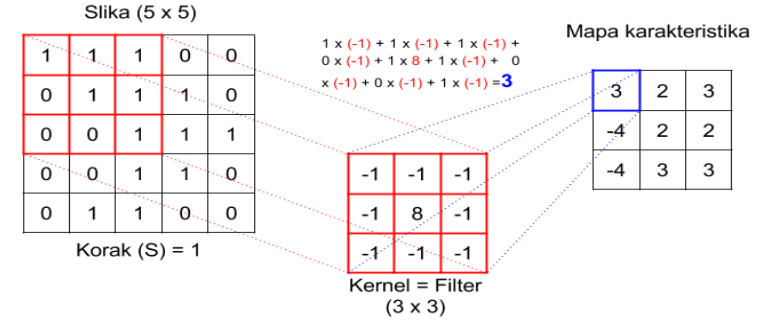


Слика 9: Могуће хиперравни (лево), оптимална хиперраван (десно) [14]

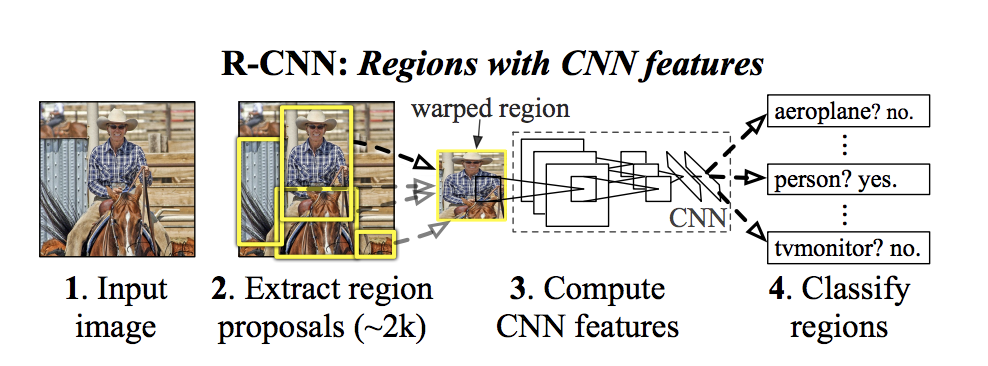
HOG фичер дескриптор у комбинацији са SVM класификатором се може користити за детекцију објеката у реалном времену и популаран је и данас. Даје добре резултате, инваријантан је на промене у величини објеката и осветљењу, али може имати слабије перформансе када су објекти оријентисани под већим углом.

### **R-CNN детектор**

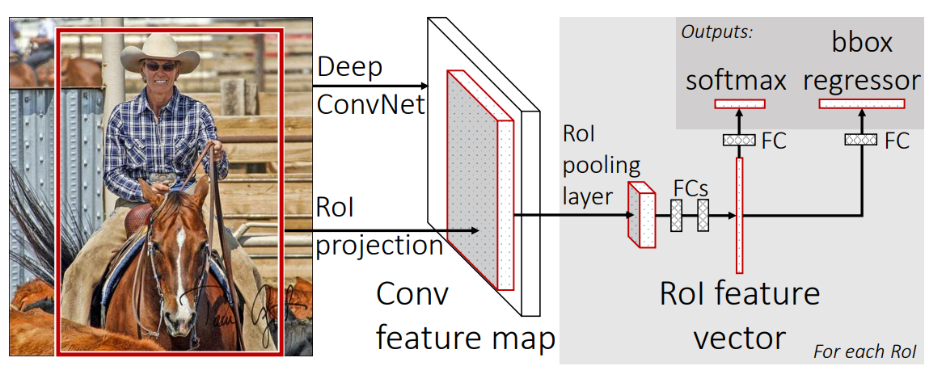
R-CNN (енг. Region-based Convolutional Neural Networks) детектор из 2013. објављен у раду [15] је један од првих који примењује дубоко учење за детекцију обреката. Заснива се на конволуционим неуронским мрежама које се по својој архитектури могу сврстати у дубоке неуронске мреже и имају велику примену у домену рачунарског вида.

Конволуционе неуронске мреже су добиле име по конволуцији, оператору који се примењује у обради слика за изоштравање и замућење, као и за детектовање ивица. Конволуциони филтери (кернели) се примењују на слике како би се извукле корисне карактеристике (фичери) и креирале њихове мапе (енг. feature maps), при чему се врши и редукција резолуције слика. Филтер се представља дводимензионалном матрицом малих димензија која се састоји од реалних вредности и примењује се у конволуционом слоју мреже на слику која се обрађује у том слоју. На слици 10 може се видети примена 3х3 филтера са кораком померања 1 на монохроматску слику димензија 5х5. У сваком кораку се вредности на истој позицији множе и сабирају са производима других парова у прозору. Архитектура мреже је таква да се на почетку налази улазни слој путем кога се слика уводи у мрежу, затим следи један или више конволуционих слојева између којих се могу наћи слојеви сажимања (eng. pooling layer) који се користе са циљем прогресивног смањења слике, слој активационе функције (најчешће ReLU) који може значајно побољшати перформансе мреже и који се може убацити након сваког конволуционог слоја, и на крају, опционо, један или више потпуно повезаних (eng. Fully Connected – FC) слојева. [16]

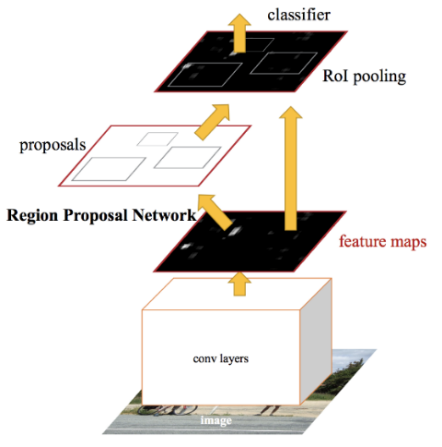
Слика 10: Примена конволуционог филтера [16]

На слици 11 је приказано како R-CNN модел функционише. Систем за детекцију објеката се састоји из три модула. Први модул применом селективног тражења издваја из улазне слике око 2000 региона који представљају кандидате за детекцију објеката. Ови региони су у суштини правоугаоници за које постоји вероватноћа да садрже објекат (без информације о томе којој класи припада тај објекат). Алгоритам селективног тражења генерише сегменте слике који могу садржати објекат на основу боје, текстуре и облика. Други модул је конволуциона неуронска мрежа (позната као “AlexNet”) са 5 конволуционих слојева и 2 потпуно повезана слоја која издваја фичере (векторе фичера фиксне дужине од 4096 елемената) из сваког предложеног региона. Трећи модул је скуп SVM-а за сваку појединачну класу уз помоћ којих се класификују региони. Након класификације, примењује се пост-процесирање да се пречисте оквирни правоугаоници и елиминишу дупликати (bounding box регресија). Овај модел има значајно боље перформансе од HOG детектора. [15]

Слика 11: Начин функционисања R-CNN модела [15]

Процеси тренирања мреже и тестирања R-CNN модела трају дуго, због чега је он веома спор. Fast R-CNN из 2015. године је побољшање R-CNN метода у смислу брзине тренирања и тестирања, као и тачности детекције. Користи се VGG16 неуронска мрежа која је 9 пута бржа од мреже коришћене у R-CNN. На слици 12 приказана је архитектура Fast R-CNN-а. Мрежа која као улаз има комплетну слику и скуп предложених региона процесира слику и генерише конволуциону мапу фичера. Дакле, генерише се једна мапа фичера, а не 2000. Затим, за сваки предложени регион, RoI (region of interest) pooling слој издваја вектор фичера фиксне дужине из мапе фичера. Вектор фичера се прослеђује низу потпуно повезаних слојева (FCs) који се гранају у 2 излазна слоја: један који генерише softmax вероватноћу над K класа објеката (+1 за позадинску класу) и други који генерише позиције оквирних правоугаоника за сваку од К класа. [17]

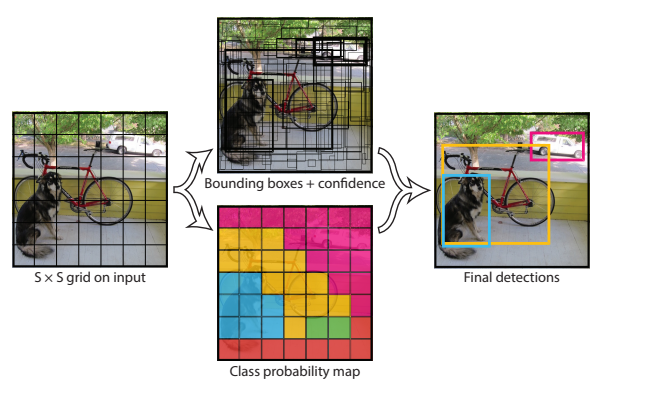
Слика 12: Архитектура Fast R-CNN-а [17]

Претходна два алгоритма користила су селективну претрагу за проналажење региона која је веома спора, односно представља уско грло у систему. Уместо овог алгоритма, Faster R-CNN користи дубоку конволуциону мрежу која предлаже регионе. На слици 13 се може видети како изгледа архитектура комплетног система. Један модул је RPN (Region Proposal Network), мрежа која на основу конволуционе мапе фичера издваја предложене регионе, а други модул на основу исте мапе фичера и предложених региона ради класификацију. RPN је потпуно повезана конволуциона мрежа која симултано предвиђа границе објеката и њихов objectness score. Систем је јединствена неуронска мрежа за детекцију објеката. [18]

Слика 13: Faster CNN алгоритам [18]

Mask R-CNN детектор из 2017. године је унапређење Faster R-CNN. Разлика између њих је у томе што Mask R-CNN додаје грану за предикцију маске објекта паралелно са постојећом граном за препознавање оквирног правоугаоника. Маска која се рачуна на нивоу пиксела нам омогућава да раздвојимо објекат од позадине примењујући сегментацију инстанци (енг. instance segmentation). [19] Још једно побољшање Fast RCNN и Faster RCNN детектора, из 2021. године, је G-RCNN (granulated RCNN) који издваја регионе од интереса примењујући концепт грануларности у дубоким конволуционим неуронским мрежама. [20]

### **YOLO детектор**

YOLO детектор је презентован у раду [21] који је објављен 2016. године и представљао је нови приступ детекцији објеката. За разлику од детектора који у основи имају класификатор и примењују детекцију над различитим деловима и величинама слике, детекција објеката је посматрана као проблем регресије просторно одвојених оквирних правоугаоника и њима придружених вероватноћа. Неуронска мрежа предвиђа оквирне правоугаонике и одговарајуће вероватноће директно из комплетних слика једним погледом на слику - одатле и потиче назив детектора („погледај само једном“).

Слика 14: Модел YOLO детектора [21]

Модел детектора приказан је на слици 14. Систем дели улазну слику на грид димензија S×S. Ако центар објекта упада у ћелију грида, та ћелија ће бити задужена за његову детекцију. Свака ћелија предвиђа B обухватајућих правоугаоника. Сваки обухватајући правоугаоник има 5 предикција: x, y, w, h, и confidence. Координате (x, y) представљају центар правоугаоника релативно у односу на границе ћелије грида. Ширина и висина (width и height) представљене су релативно у односу на целу слику. Поузданост (confidence) се рачуна као , где је вероватноћа да правоугаоник садржи објекат, а IOU (intersection over union) осликава тачност граница правоугаоника (представља количник површине преклапања предиктованог правоугаоника и било ког ручно означеног правоугаоника из тест скупа података који специфицира где се објекат налази, и површине уније та два правоугаоника). Свака ћелија грида такође предвиђа C условних вероватноћа за класе . Ове вероватноће представљају вероватноћу појављивања објеката сваке од класа у грид ћелији. Предвића се само један скуп вероватноћа по ћелији независно од броја обухватајућих правоугаоника B. Приликом тестирања множе се условне вероватноће припадања класи и индивидуалне вероватноће за сваки обухватајући правоугаоник, што нам даје вероватноћу припадања класи за сваки обухватајући правоугаоник:

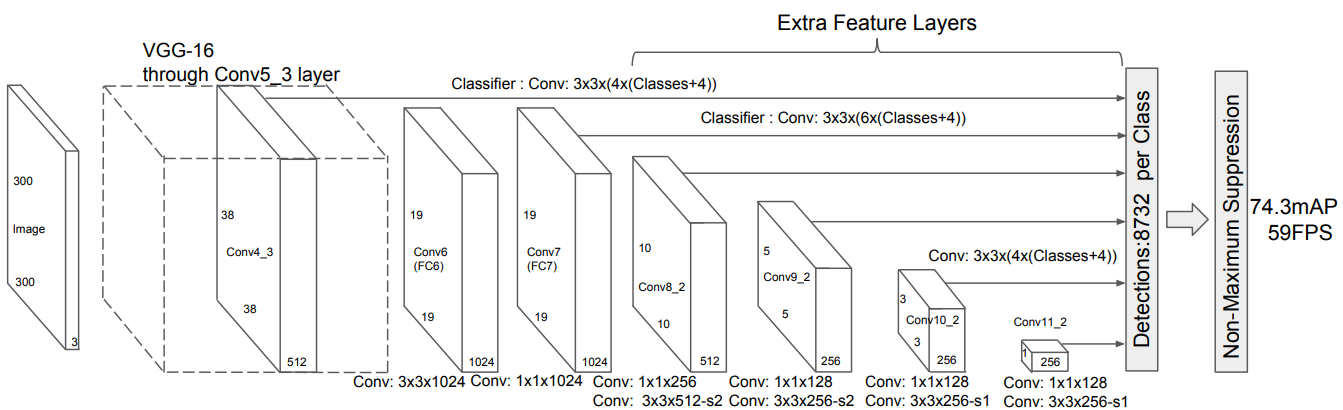
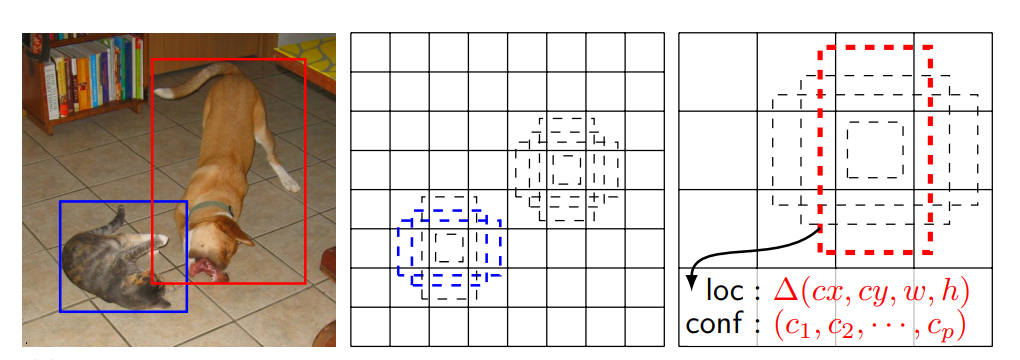
.

Ово нам даје вероватноћу да се класа i појављује у правоугаонику, као и информацију о томе колико добро правоугаоник представља објекат. Предикције се кодирају као S×S×(B ∗ 5 + C) тензор. Модел је имплементиран као конволуциона неуронска мрежа која има 24 конволуционих слојева и 2 потпуно повезана слоја, док бржа варијанта мреже има мање конволуционих слојева (9) и мање филтера у тим слојевима. Иницијални конволуциони слојеви издвајају фичере из слике, док потпуно повезани слојеви предвиђају излазне вероватноће и координате. Архитектура је веома брза, основни модел обрађује у реалном времену 45 фрејмова у секунди на Titan X GPU, док мања верзија мреже Fast YOLO обрађује преко 155 фрејмова у секунди, задржавајући дупло већи mAP (mean Average Precision) од осталих детектора који раде у реалном времену. [21]

Главни недостаци овог алгоритма су што не детектује добро мале објекте и објекте који су блиско груписани зато што дели слику на грид у коме је свака ћелија задужена за детекцију једног објекта, тако да ако постоји више објеката у једној ћелији они неће бити детектовани. Са друге стране, има знатно мање лажних предикција када нема ничега на слици. У наредним годинама су објављиване верзије овог детектора од којих је последња YOLOv7, објављена ове године. YOLOv7 се сматра најбољим детектором у овом тренутку.

### **SSD детектор**

SSD (Single Shot MultiBox Detector) детектор је представљен у раду [22] 2016. године. Детекцију објеката обавља коришћењем дубоке конволуционе неуронске мреже једним проласком кроз мрежу и дизајниран је тако да ради у реалном времену.

Архитектура детектора приказана на слици 15 састоји се из два дела и заснована је на VGG-16 мрежи која се користи за класификацију слика (уместо ње могу се користити и друге мреже као основа). Процес детекције обухвата издвајање мапа фичера и примену конволуционих филтера како би се детектовали објекти. VGG-16 мрежа је одсечена пре класификационог слоја и њени потпуно повезани слојеви 6 и 7 су конвертовани у конволуционе слојеве (Conv6 и Conv7). Затим следи скуп додатних конволуционих слојева (Conv8\_2, Conv9\_2, Conv10\_2 и Conv11\_2), при чему се резолуција слојева прогресивно смањује. Почетни конволуциони слојеви доприносе детекцији мањих објеката, а крајњи детекцији већих објеката. У конволуционом слоју Conv4\_3 слика се дели на 38×38 ћелија, при чему се за сваку од њих генеришу 4 подразумевана правоугаоника (енг. default boxes), а за сваки од њих се генерише по једна предикција, тако да имамо укупно 38×38×4 предикција. Генерисање више предикција се зове multibox. Предикција се састоји од предиктованог оквирног правоугаоника и 21 вредности које означавају припадност свакој од класа (20 класа за објекте и једна класа за позадину, тј. случај кад нема објекта). Предиктован оквирни правоугаоник је дефинисан релативно у односу на подразумевани правоугаоник коришћењем 4 вредности: (∆cx, ∆cy, ∆w, ∆h) које представљају офсете у односу на центар (cx, cy), ширину и висину подразумеваног правоугаоника. За класу објекта узима се класа са највећом вредношћу. Рачунање предикција се обавља применом 3×3 конволуционог филтера над сваком ћелијом. For example, in Conv4\_3, we apply four 3 × 3 filters to map 512 input channels to 25 output channels (21 scores for each class plus one boundary box). Овај поступак се обавља у сваком конволуционом слоју. Са смањењем величине слојева смањује се и резолуција мапа фичера, што омогућава детекцију објеката различитих величина и значајно побољшава тачност детектора. Укупно има 6 конволуционих слојева, при чему се у некима од њих генерише 4, а у некима 6 предикција. SSD укупно генерише 8732 (38\*38\*4 + 19\*19\*6 + 10\*10\*6 + 5\*5\*6 + 3\*3\*4 + 1\*1\*4) предикција. SSD дефинише фактор скалирања за сваки од слојева. Комбиновањем ових фактора скалирања и жељених односа ширине и висине израчунавају се ширина и висина подразумеваних правоугаоника. На нивоу једног слоја користе се исти подразумевани правоуганици за сваку ћелију и они су центрирани у тој ћелији, али различити слојеви користе различите подразумеване правоуганике услед разлике у резолуцији.

Слика 15: Архитектура SSD детектора [22]

Слика 16: Слика са GT правоугаоницима (лево), 8 × 8 мапа фичера (средина), 4 × 4 мапа фичера (десно) [22]

Предикције се могу класификовати као позитивна и негативна поклапања. Ако одговарајући подразумевани правоугаоник има IoU већи од 0.5 са ground truth (GT), поклапање се сматра позитивним, а у супротном негативним. GT правоугаоници су они којима су ручно означени објекти на тренинг сликама (слика 16 лево). На слици 16 у средини имамо два подразумевана правоугаоника поклопљена са мачком и један са псом (десно), који се сматрају позитивним преклапањима, а остали правоугаоници се сматрају негативним. На слици 16 је приказан и пример како SSD детектује објекте различитих величина и односа ширина/висина. Пас одговара једном подразумеваном правоугаонику (црвеном) у слоју са 4×4 мапом фичера, али ниједном подразуменом правоугаонику у мапи фичера веће резолуције (8×8). Мачка је мања и може се детектовати само коришћењем слоја са 8×8 мапом фичера, у 2 подразумевана правоугаоника (плави). Губитак при лоцирању (localization loss) је разлика у поклапању између GT правоугаоника и предиктованог оквирног правоугаоника, при чему се посматрају само позитивна поклапања. Губитак у поузданости (confidence loss) је грешка услед предикције класе, при чему се посматрају и позитивна и негативна поклапања. Финална loss function се израчунава као тежинска сума localization loss и confidence loss. Како се генерише много више предикција него што је присутно објеката, има много више негативних него позитивних поклапања, што креира дисбаланс међу класама и утиче на процес тренирања. Уместо коришћења свих негативних поклапања, она се сортирају по израчунатом confidence loss и бирају се она са највећим губитком, при чему се одржава однос између позитивних и негативних на 3:1. Још једна важна ствар за побољшање тачности је повећање скупа података (енг. data augmentation) коришћењем flipping, cropping, and color distortion. На крају, користи се non-maximum како би се уклониле предикције дупликати. Предикције се сортирају по поузданости и задржава се најбољих 200 предикција по слици, док се елиминишу предикције са поузданошћу мањом од 0.01 и IoU мањим од 0.45. [23]

SSD даје слабије резултате за мале објекте зато што они могу бити детектовани само у слојевима највеће резолуције, а они садрже мање значајне фичере. Тачност детекције расте са повећањем броја подразумеваних правоугаоника, али по цену брзине.

## **Детекција и препознавање лица**

Када је реч о обради слика лица прво треба дефинисати појам детекције лица зато што је то процес који најчешће претходи било којој обради лица. Детекција лица је рачунарска техника која се користи за проналажење људског лица на дигиталним сликама. Може се сматрати спефицичним случајем детекције објеката и односи се на испитивање да ли се на слици налази лице, где је лоцирано и које је величине. [24] Као најпознатији алгоритми који се користе за детекцију лица могу се издвојити Viola-Jones и HOG заједно са Linear SVM (описани у претходном поглављу), Eigenfaces и методе засноване на неуронским мрежама. Алгоритми се генерално разликују по понашању при екстремним условима као што су лоше осветљење, различити положаји лица, изрази лица, лица веома мале или велике резолуције, прекривеност лица (нпр. наочаре, коса, брада), комплексност позадине (присуство великог броја објеката) итд. Након детекције лица могу се издвојити карактеристичне тачке лица (енг. facial landmarks) које означавају истакнуте деловe лица попут очију, обрва, носа, усана и образа. Карактеристичне тачке се успешно примењују нпр. за поравнање лица, замену лица, препознавање лица, одређивање положаја главе, праћење покрета очију.

Препознавање или идентификација лица је процес утврђивања коме одређено лице са слике припада, односно, упоређивање неког лица са познатим лицима из базе података у циљу проналажења поклапања. [25] Не мора се у свим применама радити упоређивање са лицима из базе података јер је некада потребно само дозволити приступ одређеној особи, а осталима забранити (као на пример код откључавања телефона). Препознавању најчешће претходи детекција лица како би се обрађивао само део слике који представља лице. Постоје и 3D системи за препознавање лица који укључују додатне информације о облику лица у простору и они могу дати боље резултате него 2D системи. У циљу побољшања резултата препознавања лица често се врши поравнање лица, при чему неке методе посматрају 3D модел лица, а једноставније методе се ослањају само на карактеристичне тачке (конкретно, на тачке које одређују очи).

Код система за препознавања лица може бити велики проблем уколико неко злонамерно покуша да приступи систему представљајући се као неко други. Када се препознавање врши на основу података са камере, може се десити да неко подметне слику или видео друге особе и, уколико нема провере да ли је лице на камери реално („живо“), систем подметнута лица може препознати као валидна. Постоје различити приступи овом проблему, а неки од њих су анализа текстура, анализа фреквенци (Фуријеов домен лица), анализа променљивих величина између узастопних фрејмова, алгоритми базирани на хеуристикама (покрети очију, усана, трептање), алгоритми оптичког тока (разлике у оптичким токовима генерисаним од стране 3D и 2D равни), посматрање 3D облика лица, или комбинација претходно наведених приступа. [26]

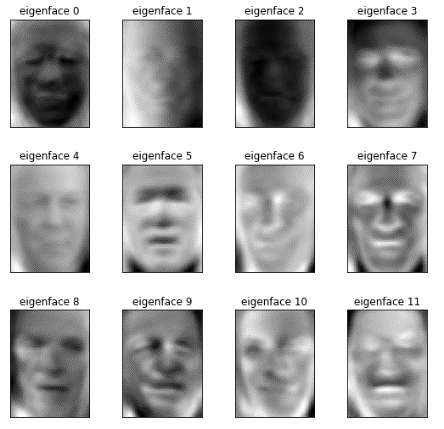
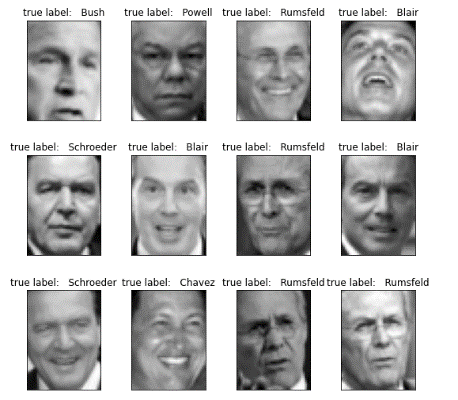
Прво су били развијени системи који користе геометрију лица за идентификацију (позиција и величина очију, носа, образа и браде), затим системи који користе алгоритме машинског учења (издвајање фичера и тренирање класификатора), а у скорије време за препознавање лица све више се користе алгоритми дубоког учења. Један од најзначајнијих алгоритама је Eigenfaces, где је коришћена техника линеарне алгебре за редукцију димензија која се зове Principal Component Analysis (PCA). Затим је представљена метода Fisherfaces која користи Linear Discriminant Analysis (LDA). Појавила се и метода базирана на фичерима, Local Binary Patterns (LBP), која се и данас користи у многим апликацијама. Када је у питању дубоко учење, постоје специјалне архитектуре које се зову *siamese networks*. FaceNet и OpenFace су једни од најпопуларнијих модела дубоког учења који се користе за препознавање лица. [27]

### **Eigenfaces**

У раду [28] из 1991. године представљен је Eigenfaces алгоритам за детекцију и препознавање лица који се заснива на редукцији димензија простора слика лица коришћењем PCA (енг. Principal Component Analysis).

Алгоритам најпре обрађује скуп од М тренинг слика лица које су сиве, исте величине и поравнате тако да се кључни делови лица што више поклапају. Посматрамо слике димензија N×N (уопштем случају могу бити произвољне величине) које се могу линеаризовати и представити као вектор димензија N2, односно као тачка у простору са N2 димензија. Израчунава се средње лице које је вектор дужине N2 чији су елементи просечна вредност одговарајућих елемената вектора тренинг слика, након чега се свака тренинг слика представља као разлика оригиналне слике и слике средњег лица. Овиме заправо померамо средњу тачку (средње лице) у координатни почетак како би се лакше израчунала коваријанса. Варијанса представља одступање вредности тачака од средње вредности и рачуна се за сваку од димензија, а коваријанса представља меру повезаности димензија. Матрица коваријансе има димензије које одговарају броју димензија простора са којим радимо, у нашем случају N2×N2 иможе се израчунати као , при чему је матрица А добијена смештањем вектора тренинг слика у колоне. У пресеку неке врсте и колоне налазиће се вредност коваријансе димензија које су везане за ту врсту и колону. Циљ примене PCA је пронаћи принципалне компоненте које најбоље описују дистрибуцију слика лица у тренинг скупу, односно eigenvector-е матрице коваријансе која одговара скупу тренинг слика. Eigenvector је вектор димензија N2 и може се посматрати као лице које личи на духа и које називамо „еigenface“ или „својствено лице“. Рачунање матрице коваријансе може се ефикасније обавити множењем АT и А, чиме се добија матрица димензија М×М, при чему важе следеће релације:

где су својствене вредности обе матрице, својствени вектори матрице , а својствени вектори матрице . На основу израчунатих вектора могу се добити вектори коришћењем једнакости . Свако лице из тренинг скупа може бити представљено као линеарна комбинација К својствених лица која најбоље репрезентују варијације међу лицима и која описују К-димензиони простор који називамо „простор лица“, при чему је К много мање од иницијалне димензионалности простора слика лица. Тренинг слике се могу представити вектором тежина које се јављају у линеарној комбинацији. На слици 17 лево су приказана тренинг лица, у средини је представљено средње лице, а десно је приказано првих неколико eigenface репрезентација. Светлији региони одговарају већим варијацијама, а тамнији мањим.



Слика 17: Тренинг лица (лево), средње лице (средина), еigenfaces (десно) [29]

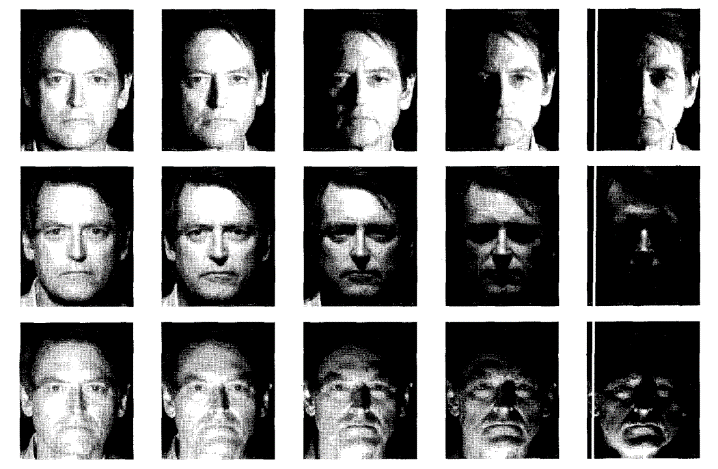


Када се анализира нова слика лица, она се модификује слично као тренинг слике тако што се од ње се одузима средње лице, а затим се пројектује на К-димензиони простор као линеарна комбинација својствених лица. За потребе детекције лица, из К-димензионог простора се реконструише слика (тако што се средњем лицу дода сума производа својствених лица и одговарајућих тежина придружених пројектованој слици) и пореди са оригиналном сликом. У случају да је та разлика већа од граничне вредности, на слици нема лица, а у супротном је лице детектовано. Уколико је циљ препознавање лица, онда се добијени К-димензиони вектор пореди (рачуна се Еуклидско растојање) са векторима добијеним пројекцијом тренинг слика. Што је мање растојање, лица су сличнија. Лице се идентификује као лице са којим има најмање растојање. [29]

Предност Eigenfaces алгоритма је што није рачунски захтеван и брзо се извршава. Један од недостатака овог алгоритма је што захтева поравнање слика лица приликом тренирања и препознавања зато што се ради на нивоу пиксела, тако да је потребно да се карактеристике лица скоро савршено поклапају. Такође, захтева фронтална лица и осетљив је на промене у осветљењу.

### **Fisherfaces**

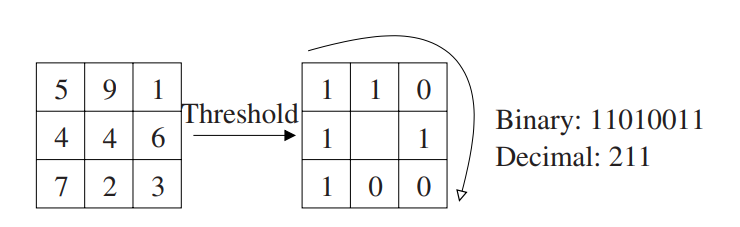
Fisherfaces метод представљен 1997. године у раду [30] је уз Eigenfaces један од најпопуларнијих алгоритама препознавања лица. Овај алгоритам се такође заснива на редукцији димензија, али користи линеарну дискриминантну анализу (LDA). Сваки пиксел на слици посматран је као тачка у високодимензионом простору и слика је линеарно пројектована у нискодимензиони потпростор који није осетљив на промене у осветљењу и експресији лица.

Eigenfaces користи пројекције које максимизују разлике између лица у целом скупу слика, што не даје добре резултате при препознавању када постоје промене у осветљењу јер су веће разлике између слика истог лица при различитом осветљењу него између слика лица различитих особа, што се може уочити на слици 18. Fisherfaces користи пројекцију која ће максимизовати однос расподеле између класа и расподеле унутар класа. Тежи се максимизацији удаљености лица различиитх класа, а минимизацији удаљености унутар сваке класе. Рачунају се матрице коваринсе између класа и унутар класа, а затим својствени вектори и својствене вредности. Бирају се вектори са највећом својственом вредношћу. Као код Eigenfaces алгоритма, тренинг слике се пројектују у потпростор слика, улазна слика се такође пројектује у овај потпростор и упоређују се растојања. [30]

Слика 18: Слике лица при различитом осветљењу [30]

Метод је дао боље резултате од Eigenfaces алгоритма и није осетљив на велике варијације у осветљењу (које се односе на промену у интензитету, али и смеру и броју светлосних извора) и експресији лица. Један од недостатака алгоритма је што не подржава промене у пози и, слично Eigenfaces алгоритму, захтева да слике лица буду поравнате.

### **LBP**

Алгоритам који користи Local Binary Patterns (LBP) за препознавање лица приказан је у раду [31] који је објављен 2006. године. Метод узима у обзир информације о облику и текстури приликом представљања слике. LBP оператор сваки пиксел на слици пореди са околним пикселима и резултат посматра као бинарну вредност. Пример примене оператора дат је на слици 19. Вредностима већим од 4 придружена је 1, а мањим од 4 придружена је 0, након чега се добијена осмобитна бинарна вредност преводи у децимални број. Затим се хистограм ових вредности користи као дескриптор текстуре. Први корак у алгоритму је превођење слике у сиву и подела на матрицу од 7×7 једнаких ћелија. Затим се за сваку од ћелија израчуна LBP хистограм фичера. Ако се посматра 8 суседних пиксела, хистограм је дужине 256. Рачунањем хистограма за сваку ћелију заправо енкодирамо просторне информације као што су очи, нос, уста... Неки региони слике носе више информација, неки мање, тако да се уводе тежине које се придружују ћелијама. На слици 20 лево је приказано лице подељено на ћелије, а десно тежинска шема за сваку ћелију. Бели квадрати (очи) имају тежину 4 (њихови хистограми су помножени са 4), светло сиви тежину 2 (уста и уши), тамно сиви 1 (унутрашњи образи и чело), док црни имају тежину 0 (нос и спољашњи део образа).

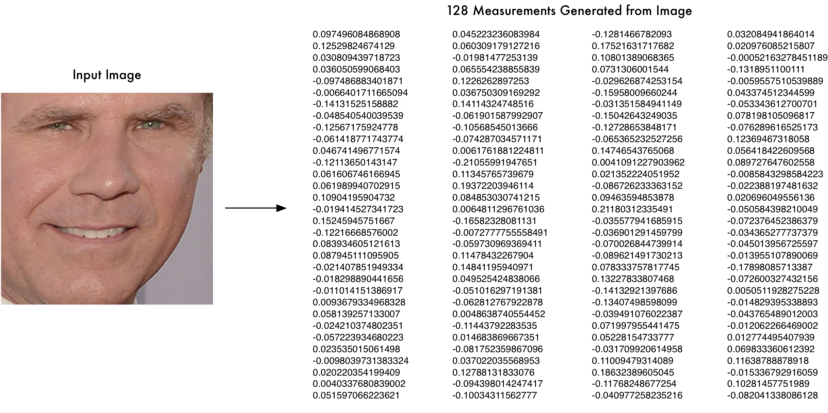
Слика 19: LBP оператор [31]

Слика 21: Слика лица подељена на ћелије (лево), тежинска шема за одговарајуће ћелије (десно) [31]

Вредности тежина су експериментално утврђене. Тежински хистограми се надовезују и формирају јединствени хистограм фичера који презетује слику лица. Препознавање лица се врши упоређивањем растојања. Улазно лице се обрађује као и тренинг лица (издвајају се LBP, додају им се тежине, надовезују се) и затим се примени k-NN (k-Nearest Neighbors) алгоритам (k=1) са растојањима како би се нашло најближе лице из тренинг података. [31]

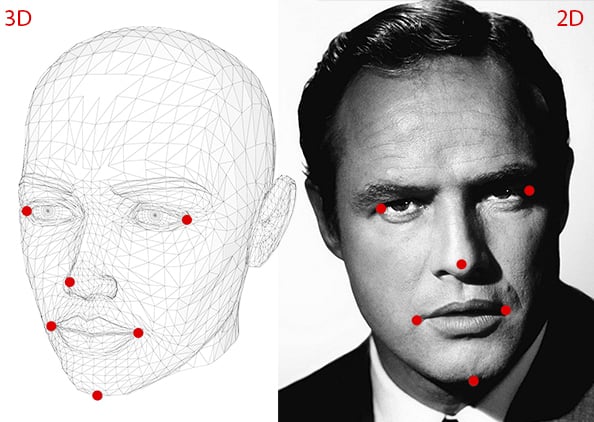
Овај алгоритам може да се ажурира приликом додавања новог лица у систем, док неки други алгоритми захтевају да сва лица која ће се идентификовати буду присутна у време тренирања. LBP алгоритам је отпорнији на шумове јер не ради директно са интензитетима пиксела и углавном даје боље резултате него Eigenfaces алгоритам. Ефикасан је и омогућава веома брзо издвајање фичера.

### **Дубоко учење**

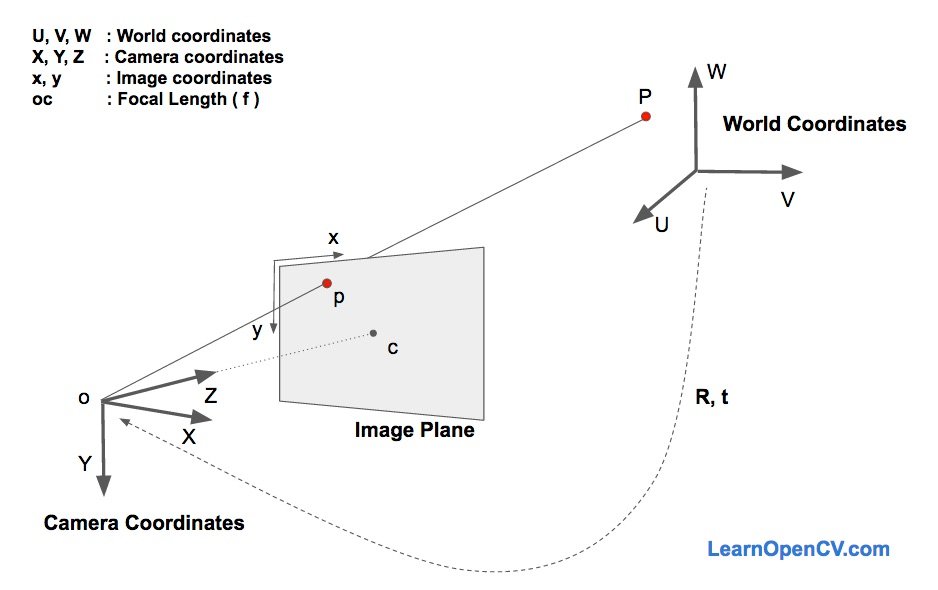
Дубоке конволуционе неуронске мреже могу се користити за препознавање лица са великом тачношћу. На почетку треба детектовати лице, издвојити карактеристичне тачке и поравнати га коришћењем афиних трансформација тако да очи и уста буду увек на истим позицијама на слици, како би било олакшано упоређивање. За разлику од неуронских мрежа за детекцију објеката које се тренирају да би препознале објекат на слици, ове неуронске мреже се тренирају да генеришу 128 мера (низ реалних бројева) које карактеришу свако лице (слика 21). Тренирање мреже се обавља тако што се користе 3 различите слике (енг. *triplets*), при чему две припадају истој особи, а трећа некој другој особи. Тежине мреже се подешавају тако да слике исте особе имају врло сличне векторе, а слике те особе и неке друге што различитије. Овај поступак се понавља милионима пута за милионе слике да би мрежа научила да поуздано генерише мере за сваку особу. Када се мрежа једном истренира може се користити за генерисање вектора за било које лице, па и потпуно непознато. Да би се непознато лице препознало треба наћи у бази познатих лица оно које има најближе мере са непознатим. [32]

Слика 21: Пример 128 мера лица које генерише неуронска мрежа [32]

## **Одређивање положаја главе**

Поза неког објекта се посматра као његова релативна оријентација у односу на камеру. Проблем естимације позе се често назива Perspective-N-Point проблем или PNP. Поза објекта се одређује када имамо калибрисану камеру и знамо локације N 3D тачака које одговарају 2D пројекцијама на слици.

Слика 22: 6 карактеристичних 3D тачака и одговарајућих 2D тачака [33]

Објекат се може кретати у односу на камеру применом транслације или ротације, а то кретање можемо посматрати и као кретање камере око објекта. Транслацијом се назива померање камере са њене тренутне 3D позиције (X, Y, Z) на нову 3D позицију (X’, Y’, Z’) и она се може представити вектором t = (X’ – X, Y’ – Y, Z’ – Z). Ротацијом се назива ротација камере око X, Y и Z осе и може се представити на више начина: коришћењем Ојлерових углова, 3x3 ротационе матрице или смером ротације и углом. За одређивање позе главе потребно је знати 2D координате (x, y) неколико тачака на слици (врх носа, врх браде, углови очију и углови усана), 3D координате истих тачака (слика 22), као и унутрашње параметре камере (фокалну дужину камере, оптички центар слике и параметре радијалне дисторзије). Уместо 3D модела лица може се користити генерички 3D модел са тачкама у произвољном (светском) координатном систему: врх носа (0.0, 0.0, 0.0), брада (0.0, -330.0, -65.0), леви угао левог ока (-225.0, 170.0, -135.0), десни угао десног ока (225.0, 170.0, -135.0), леви угао уста (-150.0, -150.0, -125.0), десни угао уста (150.0, -150.0, -125.0). Оптички центар слике се може апроксимирати центром слике, фокална дужина ширином слике и може се претпоставити да радијална дисторзија не постоји. Посматрамо 3 координатна система приказана на слици 23: светски координатни систем, координатни систем камере и координатни систем слике. 3D координате неколико тачака лица представљене су у светским координатама и ако знамо ротацију и транслацију можемо их трансформисати у 3D тачке у координатама камере. Ове тачке могу бити пројектоване на раван слике коришћењем унутрашњих параметара камере чиме се добијају тачке у координатном систему слике.

Слика 23: Координатни системи [33]

На слици 23 о је центар камере, а раван која је приказана је раван слике. Нас интересују једначине које дају пројекцију p 3D тачке P на раван слике. Нека је (U, V, W) позиција тачке P у светским координатама. Ако знамо ротациону матрицу R (3×3) и транслациони вектор t (3×1), можемо израчунати (X, Y, Z) координате тачке P у координатном систему камере на следећи начин:

,

односно:

.

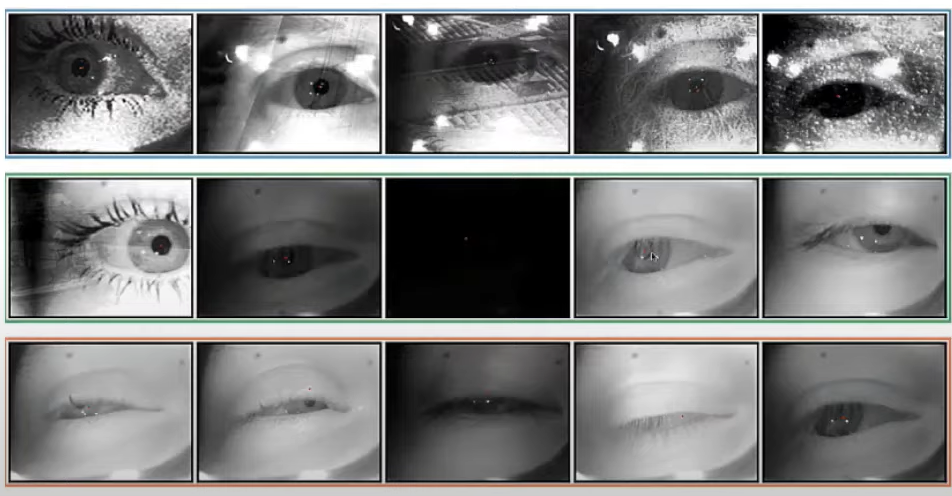
Ако имамо довољан број парова (X, Y, Z) и (U, V, W), можемо решити систем линеарних једначина и добити непознате и (, , ). Нама су познате тачке у 3D моделу (U, V, W), али не знамо (X, Y, Z), већ само 2D тачке (x, y). У одсуству радијалне дисторзије координате (x, y) тачке pдате су једначином:

где су и фокалне дужине у x и y смеровима, а (, ) је оптички центар. Непознати фактор скалирања *s* постоји у једначини услед чињенице да не знамо дубину слике. Ако спојимо било коју 3D тачку P са центром камере, тачка p, у којој зрак пресеца раван слике је пројекција тачке P. Све тачке дуж зрака када се споје са центром камере имаће исту пројекцију на раван слике, тако да коришћењем претходне једначине можемо добити само (X, Y, Z) са скалирањем s. Сада наша једначина изгледа овако:

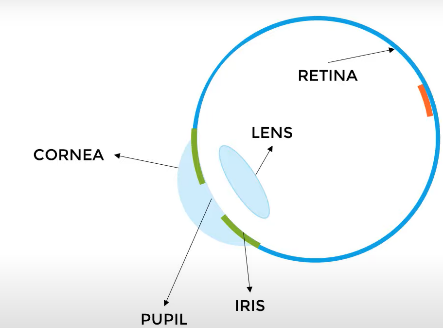
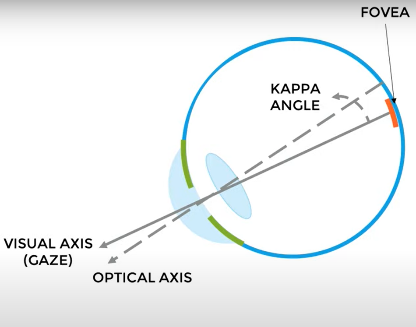
и може се решити коришћењем метода који се зове директна линеарна трансформација [(DLT)](https://en.wikipedia.org/wiki/Direct_linear_transformation). Овај метод се може користити кад год имамо проблем где је једначина скоро линеарна, али је помножена непознатим фактором. У идеалном случају, желимо да минимизујемо **грешку пројекције**. Ако знамо тачну позу (R и t), можемо предвидети 2D локације 3D тачака са слике пројектовањем 3D тачака на 2D слику. Како знамо 2D карактеристичне тачке лица, можемо посматрати растојање између пројектованих 3D тачака и 2D карактеристичних тачака лица. Када процењена поза буде савршена, 3D тачке пројектоване на слику ће се поклопити са 2D карактеристичним тачкама. Овај метод се може побољшати итеративном променом вредности R и t тако да се грешка пројекције смањује: [Levenberg-Marquardt](https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm) оптимизација. OpenCV библиотека нуди функције које решавају PnP проблем и могу се користити за одређивање позе. [33]

## **Праћење покрета очију**

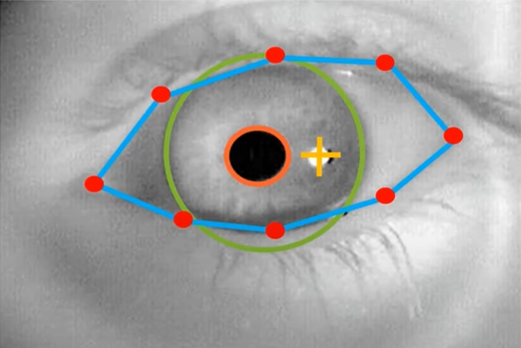
Праћење покрета очију је добро познат проблем у рачунарском виду и постоје решења која се успешно примењују за различите намене попут управљања показивачем на екрану очима, помоћи људима са одређеним видовима инвалидитета и анализе пажње.

Разликујемо два случаја праћења погледа: када желимо да одредимо тачку на екрану у коју особа гледа и када желимо да одредимо тачку у сцени у коју особа гледа. У оба случаја користи се систем са две камере. У првом случају једна камера у суштини представља екран, а друга снима лице особе и најчешће је закачена за екран. У другом случају особа носи специјалне наочаре које имају камеру у средини која је усмерена ка сцени и другу камеру која је закачена за наочаре и усмерена ка очима (најчешће је ка очима усмерено и инфрацрвено светло које се користи да би се добио бољи контраст на слици, а при томе не утиче на очи јер га људско око не може спознати). Један од најважнијих корака у одређивању смера гледања је детекција зенице. Проблеми који се могу јавити при детектовању зенице су да се око може снимати из различитих углова, особа може носити наочаре, могу јој очи бити делимично затворене, трепавице могу делимично прекрити око, постоје варијације у облику ока код људи, квалитет слике и осветљење могу бити лоши (слика 24).

Слика 24: Варијације у сликама очију [34]

Алгоритми за детекцију зенице могу се засновати на класичним техникама попут детекције ивица или поклапања елипсе, као и на конволуционим неуронским мрежама. Да би се одредио положај зенице потребно је осврнути се на грађу људског ока приказану на слици 25. Светлост пролази кроз отвор који се зове зеница (pupil) коју контролише дужица (iris), прелама се кроз сочиво (lens) и слика се пројектује на мрежњачу (retina). Око зенице налази се рожњача (cornea). Линија која повезује центар зенице и центар сочива се назива оптичка оса. Место на ретини где је слика најоштрија се зове *fovea* и налази се у центру жуте мрље. Линија која спаја ову тачку и центар сочива назива се визуелна оса и она представља правац погледа. Угао између ове 2 осе се назива *kappa* угао и најчешће је око 5 степени. Визуелну осу је могуће одредити на основу оптичке, пре чега је потребно урадити калибрацију за сваку особу. Калибрација подразумева да се корисник фокусира на одређену тачку, затим се одреди тачка у коју гледа и нађе разлика између локације те тачке и стварне тачке. Та информација се користи да се побољша наредна процена.

Слика 25: Грађа ока (лево), оптичка и визуелна оса (десно) [34]

Када светлосни зрак погоди рожњачу формира се одсјај у оку у виду белог кружића. Неке од карактеристика ока која могу бити значајне за одређивање тачке погледа су означене на слици 26. Ради се о центру и полупречнику зенице, локацији одсјаја рожање, центру и полупречнику дужице и контури (полигону) ока.

Слика 26: Значајне арактеристике ока [34]

Алгоритми за праћење погледа могу се поделити у неколико група: алгоритми засновани на 3D моделу, алгоритми засновани на регресији, Cros-Ratio алгоритми и алгоритми засновани на изгледу. Алгоритми засновани на 3D моделу главе и ока могу користити и RGBD камере за одређивање 3D параметара. Око и рожњача се могу моделовати као сфере, а зеница и дужица као концентрични кругови. Овај модел је најпрецизнији и добро се понаша при покретима главе и различитим светлосним условима, али је и најкомплекснији и захтева калибрацију и доста хардвера. Методи засновани на 2D регресији мапирају 2D фичере на поглед без експлицитног моделовања геометрије. Неки од оваквих метода су креирање полинома на основу фичера, Support Vector Regresssion (SVR) и неуронске мреже. Може се користити и Cros-Ratio метод где се постављају 4 светла у угловима екрана, затим камера и извор светлости који су на истој локацији тако да се добије светла зеница. Уз познавање центра зенице, рефклексије ових светала у оку и претпоставку да је рожњача равна површина, може се одредити тачка погледа. Алгоритми засновани на изгледу (енг. appearance based) имају за циљ рекреирање региона ока коришћењем 3D морфолошког модела ока. Овај метод је најмање прецизан, али и има најмање хардверске захтеве. [34]

У апликацијама за надгледање полагања тестова може се детектовати трептање и одређивати смер гледања особе која ради тест. Тачка екрана у коју особа гледа се може пратити уколико је потребно одрадити анализу пажње, али за утврђивање регуларности полагања то није неопходно, већ се може проверити само да ли гледа са стране, тј. ван екрана. Праћење треба обављати у реалном времену на видеу са стандардне веб камере јер је претпоставка да ученици у кућним условима немају никакву додатну хардверску опрему и не могу задовољити захтеве напреднијих система.

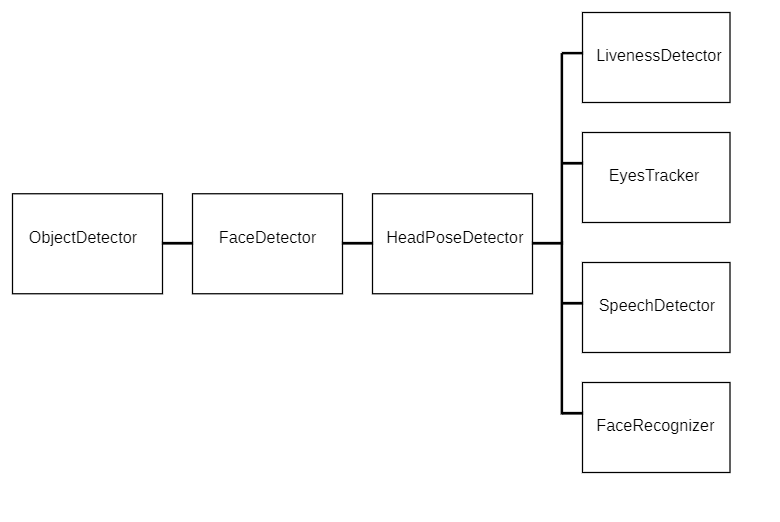
## **Детекција говора**

У зависности од потреба везаних за полагање конкретног теста у апликацијама за надгледање полагања тестова могу се вршити различите провере. Већина онлајн тестова не захтева усмене одговоре и причање током теста је забрањено, тако да може бити довољно само проверавати да ли особа прича или не (енг. voice activity detection). Могу се истовремено пратити информације са микрофона како би се детектовао људски глас и видео са камере да би се потврдило да особа која полаже тест прича. Додатно се може вршити препознавање говора уколико желимо да у извештају наведемо шта је особа изговарала (енг. speech recognition). Неки тестови могу захтевати усмене одговоре па се може вршити препознавање гласа (енг. voice recognition) како би се упоредио са гласом те особе из базе података.

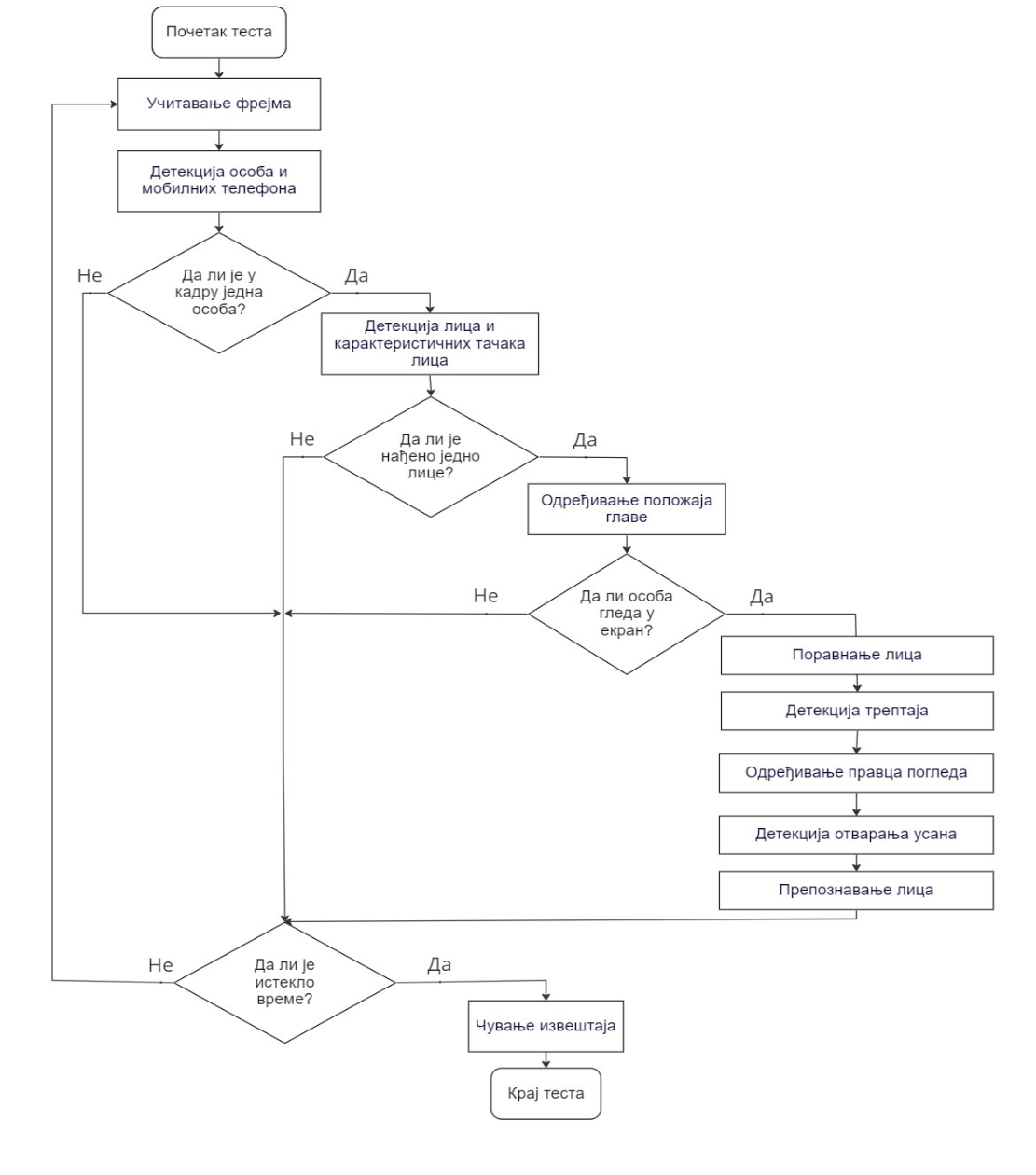
Један од начина за детекцију људског говора у улазном аудио сигналу је претворити улазни сигнал у моно формат, а затим померати прозор од 20ms над сигналом. За сваки прозор може се израчунати однос између енергије сигнала која упада у фреквентне границе говора (300 - 3 000 Hz) и укупне енергије прозора. Ако је овај однос већи од предефинисаног прага (рецимо 60%) прозор се може означити као прозор са говором. Главне предности овог приступа су што се брзо извршава и једноставан је за имплементацију, док су ограничења што има параметара које треба ручно специфицирати и што у наведеном опсегу фреквенција могу бити други звукови који нису људски глас. Још једна од метода је примена неуронских мрежа које су се добро показале када је изражено присуство шума у сигналу. [35] Када је у питању препознавање говора, већина метода користи скривени Марковљев модел (енг. Hidden Markov Model, HMM). Овај модел претпоставља да говорни сигнал, када се посматра у кратком временском интервалу (рецимо 10 ms), може бити апроксимиран стационарним процесом – процес у коме се статистичке особине не мењају током времена. Други приступ је коришћење неуронских мрежа које се најчешће употребљавају када постоји лош квалитет сигнала, бучни подаци или више говорника. У многим модерним системима се користи хибридни модел: неуронске мреже се користе да поједноставе говорни сигнал пре него што се изврши HMM препознавање. Треба узети у обзир величину речника која значајно утиче на комплексност система јер што је већи број речи теже је извршити препознавање. Такође, теже је препознати континуални говор него изоловане речи. [36] Уколико се посматра само снимак камере, без узимања у обзир аудио података, детекција говора ће бити мање поуздана, али могуће је добити прихватљиве резултате праћењем покрета усана.

# **Имплементација система за праћење регуларности полагања тестова**

За реализацију прототипа система који врши надгледање полагања онлајн тестова коришћењем камере употребљен је програмски језик *Python*, а пројекат је имплементиран у развојном окружењу *PyCharm*. Систем не представља платформу преко које се могу полагати тестови, већ само симулира полагање теста укључивањем тајмера. Надгледање полагања се обавља обрадом учитаних фрејмова са камере корисника. За рад апликације потребне су основне информације о ученицима и тестовима, а како се не би сви подаци чували у локалном фајл систему, коришћене су *Firebase* услуге. *Firebase* је Гуглова (енг. *Google*) платформа за развој апликација која нуди различите сервисе попут база података, аутентификације и аналитике. За конекцију са потребним сервисима је коришћена библиотека *firebase\_admin*, а за обраду слика (фрејмова) коришћене су библиотеке *OpenCV* и *Dlib*. Ове две библиотеке су веома популарне и користе се за потребе машинског учења и рачунарског вида. Као помоћна библиотека за нумеричка израчунавања коришћена је библиотека *NumPy*.

Кориснички интерфејс апликације је једноставан и комплетна интеракција са корисником је имплементирана преко конзоле. Омогућен је избор између три акције: додавање ученика, додавање тестова и покретање теста. Док је тест покренут на снимку са камере који је приказан ученику исписује се колико је времена остало до краја теста. На слици 27 је приказан дијаграм који илуструје компоненте система које се користе за надгледање полагања. Издваја се неколико компоненти: *ObjectDetector* који детектује одсуство особе која полаже тест и присуство више особа или мобилних телефона, *FaceDetector* који детектује лице и издваја карактеристичне тачке лица, *LivenessDetector* који проверава да ли је детектовано лице реално или је камери подметнута слика, *FaceRecognizer* који проверава да ли се детектовано лице поклапа са лицем ученика који је покренуо тест, *HeadPoseDetector* и *EyesTracker* који проверавају да ли су ученикова глава и поглед усмерени ка екрану и *SpeechDetector* који проверава да ли ученик прича. Свака од компоненти имплементирана је као посебан модул, а из главног фајла, који је улазна тачка апликације, позивају се њихове функције за детекцију и валидацију. Оваква организација кода омогућава да функционалности буду раздвојене и да касније измене у обради фрејмова буду локализоване у оквиру сваког модула. Са базом података се комуницира преко посебног модула названог *Database*.

Слика 27: Компоненте система за надгледање полагања

Подаци који се чувају о ученицима су идентификациони број, име, презиме, *e-mail* адреса, једна слика и видео снимци полагања тестова. Слике и снимци се чувају у *Firebase Storage*, а линкови до слика заједно са осталим подацима у *Firestore Database*. За тестове се чувају само информације о идентификационом броју и времену трајања. Како је у питању база података оријентисана ка документима (енг. *document-oriented*), где су подаци организовани у документе, а документи у колекције докумената, направљене су колекције *students* и *tests* у којима је сваки ученик, односно тест, представљен једним документом. Сваки документ при креирању добија аутоматски генерисан јединствени *id*. Овај *id* је искоришћен да се у складишту именује фолдер у коме ће бити чувани фајлови везани за одговарајућег ученика. Пре покретања тајмера, на основу идентификационог броја ученика учитавају се његови подаци како би му се приказала одговарајућа поздравна порука и његова слика која се користи за касније упоређивање са сликом са камере, а на основу идентификационог броја теста учитава се информација о трајању теста. Претпоставка је да сваки ученик који је убачен у систем има слику на којој му лице није ничим прекривено, гледа у камеру и не смеје се. За сваки случај се проверава да ли учитана слика садржи лице, лице се поравнава и укључује се камера.

Слика 28: Дијаграм тока обраде фрејмова

Након покретања камере, систем обрађује један по један фрејм. На слици 28 је приказан ток обраде фрејмова. На учитаном фрејму прво се детектују особе и мобилни телефони. Уколико је детектована једна особа у кадру прелази се на детекцију лица и карактеристичних тачaкa лица. Уколико је ова детекција успешна, проверава се да ли је глава особе усмерена ка екрану. Ако јесте, следи поравнање лица, након чега се даља обрада врши над поравнатим лицем. Затим се утврђује да ли се десио трептај јер се провера да ли је камери подметнута слика врши бројањем трептаја. Након тога се одређује да ли особа гледа право у екран или са стране и да ли су јој уста отворена јер се праћењем покрета усана закључује да ли особа прича. На крају се врши препознавање лица. Обрада фрејмова се прекида када прође онолико времена на колико је подешено трајање теста и тада се у видео фајл бележи извештај који се касније може прегледати ради потврђивања детектованих нерегуларности. Генеришу се два фајла: један који садржи комплетан снимак полагања са означеним нерегуларним ситуацијама, и други, који садржи само нерегуларне ситуације.

Ради лакше обраде фрејмова направљена је посебна класа која представља један фрејм и која има 4 атрибута: слику, вредност тајмера у тренутку учитавања фрејма, информацију о валидности фрејма и поруку о нерегуларностима која ће бити исписана на фрејму. Како у једном фрејму може истовремено бити више нерегуларности, сваки од детектора ажурира атрибут поруке дописивањем специфичног текста и може ажурирати атрибут слике доцртавањем додатних информација о детектованој нерегуларности. Сви учитани фрејмови се смештају у бафер главног програма на основу кога се на крају генерише видео запис полагања. Да би се у посебном фајлу издвојиле само нерегуларне ситуације, атрибут валидности сваког фрејма у коме је детектована нека нерегуларна ситуација се поставља на вредност нетачно. На крају теста, бафер фрејмова је филтриран по овом атрибуту и издвојене су све нерегуларне ситуације. Сваки од појединачних детектора има свој „прозор“ бафер у коме чува прозоре фрејмова (прозором се сматра одређени број сукцесивних фрејмова). Прозори фрејмова се посматрају како би се избегле случајне (лажне) детекције. Нерегуларна ситуација неће трајати само један фрејм, већ више узастопних фрејмова. Сви фрејмови који припадају прозору који се сматра нерегуларним биће означени као нерегуларни. У наставку ће бити детаљније описанa свака од коришћених компоненти за надгледање полагања.

## **Компоненте система**

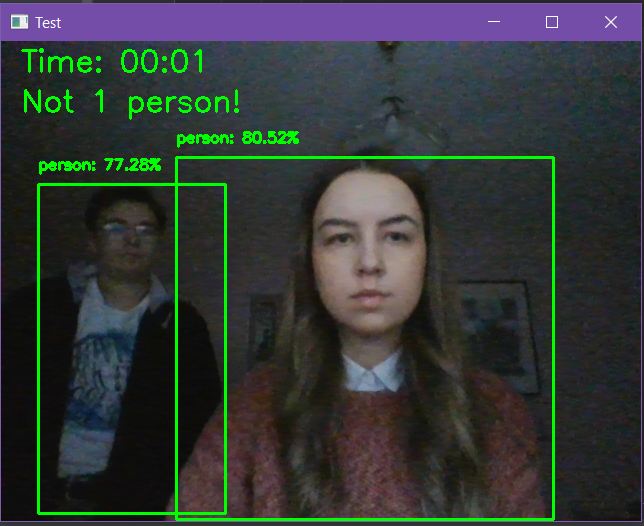
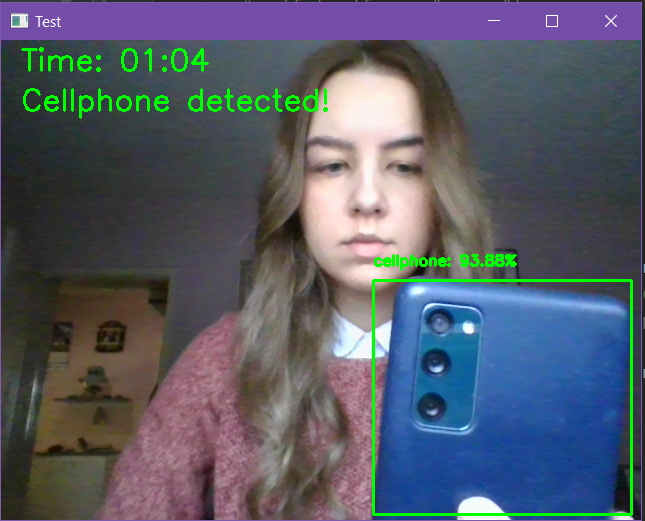
Прва компонента система је задужена за детекцију објеката и имплементирана је у оквиру модула *ObjectDetector*. При одабиру модела треба имати у виду конкретан случај коришћења и потребе саме апликације. Један од најбољих и најкоришћенијих модела је YOLO, међутим, оптимизован је за рад на GPU. Ова апликација се извршава на CPU и потребно је да има добре перформансе у реалном времену, тако да је за реализацију детекције објеката изабран MobileSSD модел. Коришћен је модул за дубоко учење из OpenCV библиотеке (доступан од верзије 3.1) који је оптимизован за извршење на CPU и који омогућава коришћење истренираних мрежа из различитих *framеwork*-a за дубоко учење као што су нпр. TensorFlow, Torch и Caffe. У пројекту је коришћена TensorFlow имплементација [37] модела који комбинује MobileNet архитектуру и SSD. MobileNet архитектуре су ефикасније у односу на друге мрежне архитектуре попут VGG или ResNet када су у питању уређаји са ограниченим ресурсима и брже се извршавају, али су мало мање тачне. Модел је трениран на COCO [38] скупу слика са 80 класа објеката који се могу срести у свакодневном животу (људи, аутомобили, животиње, предмети у кући, храна). Овај скуп садржи 330 000 слика са 1.5 милиона инстанци објеката. Како је циљ детектовати да ли постоји тачно једна особа у кадру и да ли она користи мобилни телефон, од значаја су биле класе објеката *person* и *cellphone*.

На почетку је учитан текстуални фајл који садржи лабеле за сваку класу објеката из COCO скупа слика. Овај фајл је парсиран тако да се од њега направи низ који садржи називе класа које се могу детектовати. Након тога је учитана мрежа позивом функције *readNetFromTensorflow*, која као параметре има путању до фајла који садржи топологију и тежине мреже и путању до додатног конфигурационог фајла. Величина улазне слике (фрејма) је коришћењем функције *blobFromImage* промењена на величину од 300×300 пиксела коју мрежа очекује, од сваког пиксела је одузета средња вредност пиксела тренинг слика за сваки од канала, а затим је прослеђена мрежи. Мрежа за сваки детектовани објекат враћа информацију о класи, координатама правоугаоника којим се објекат може обухватити и поузданости детекције. Координате правоугаоника су у опсегу [0, 1], те се множењем тих бројева са оригиналном ширином и висином слике добија предикцију за оригиналну слику. Само детекције које имају поузданост већу од 0.6 су разматране и међу њима су издвајане оне које припадају класама од интереса.

За сваку од класа од интереса направљен је по један бафер прозора, као и помоћни бафер који чува узастопне нерегуларне фрејмове. Тестирањем различитих величина прозора установљено је да је величина прозора од 30 фрејмова задовољавајућа. На нивоу сваког прозора се броји колико је било нерегуларних фрејмова и ако тај број прелази једну трећину укупног броја фрејмова у прозору, прозор се сматра нерегуларним. На крају сваког прозора прозор бафер се ресетује. Бафер који чува узастопне нерегуларне фрејмове у једном прозору има улогу да контролише прелаз између фрејмова. Може се десити да се нерегуларна секвенца налази на прелазу између два прозора, тј. да се један њен део налази у текућем, а други у следећем прозору, и да због ресетовања прозор бафера она не буде урачуната у потпуности (један део упада у прозор који у целини нема довољно нерегуларних фрејмова да би се сматрао проблематичним) или не буде урачуната уопште (ниједан од два прозора нема довољан број нерегуларних фрејмова да се сматрао проблематичним). Из тог разлога се на прелазу из једног у други прозор испитује да ли бафер који садржи узастопне нерегуларне фрејмове није празан. Ако није празан, текући прозор се „скраћује“ за број елемената тог бафера и не отвара се нови прозор док год има сукцесивних нерегуларних фрејмова. Када се наиђе на први валидни фрејм, испитује се да ли у баферу узастопних нерегуларних фрејмова има више од половине величине једног прозора (15) елемената и ако има ти фрејмови се означавају као нерегуларни. Након тога се прелази на следећи прозор.

Обезбеђена је и функција која ће око детектованог објекта исцртати оквирни правоугаоник и исписати колика је поузданост детекције. На слици 29 у левом делу приказан је пример детекције коришћења мобилног телефона. Треба напоменути да детекцију мобилних телефона „омета“ чињеница да ће најчешће бити држани испод стола или прекривени руком. Ако особа прислони телефон уз главу док разговара, велика је вероватноћа да он неће бити детектован, али ако га држи као на слици 29 и покушава да нешто прочита или слика питања са теста, биће детектован. У десном делу слике приказана је ситуација када су детектоване две особе у кадру, при мало слабијем осветљењу. Поузданост детекције варира, али и вредности од око 80% су довољне да нерегуларне ситуације не прођу незапажено.

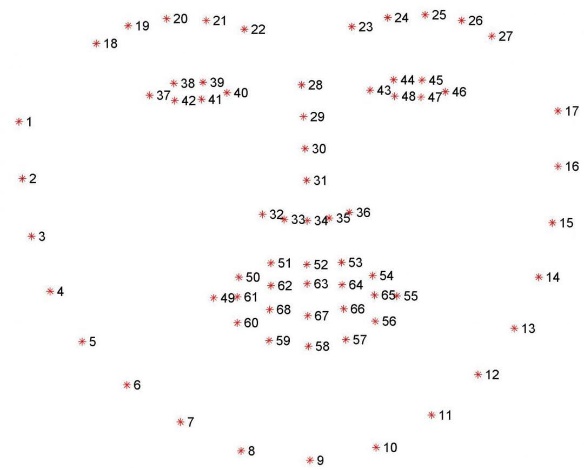
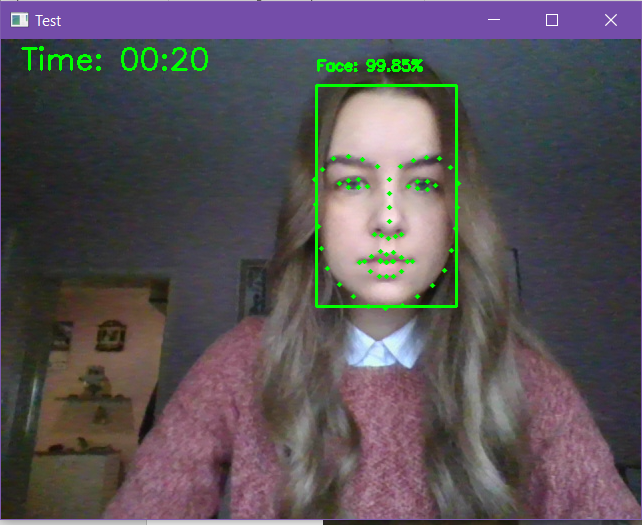
Резултат обраде сваког појединачног фрејма може утицати на преостале детекторе у систему који обрађују податке са лица зато што ако нема само једне особе у кадру, нема потребе прослеђивати тај фрејм даље - он ће представљати прекид прозора других детектора и главни програм ће позвати одговарајуће функције за њихово ресетовање.



Слика 29: Детекција мобилног телефона (лево) и детекција 2 особе (десно)

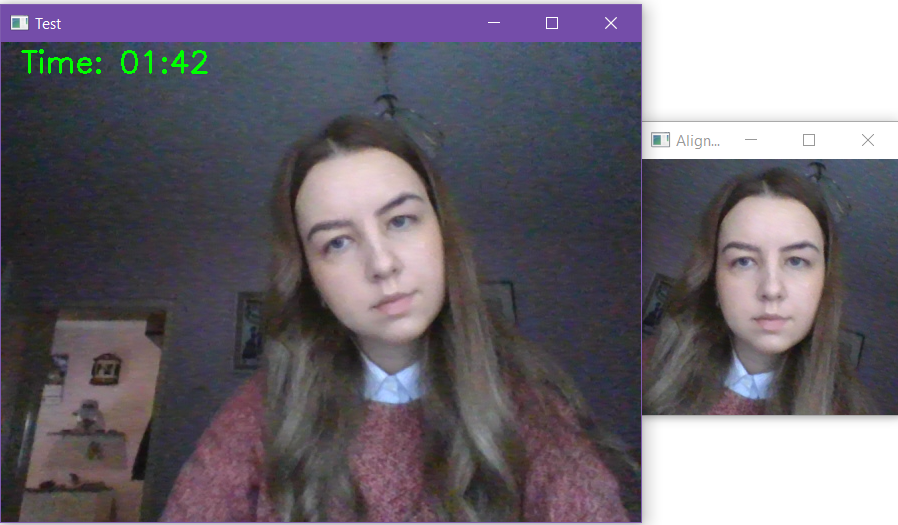
Следећа компонента задужена је за детекцију лица и карактеристичних тачака лица и имплементирана је у оквиру модула FaceDetector. При одабиру детектора лица треба размотрити неколико најзначајнијих детектора: Viola-Jones и DNN детектор из OpenCV библиотеке, а из Dlib библиотеке HOG у комбинацији са линеарним SVM и CNN детектор. Viola-Jones је напознатији детектор лица, веома је брз, нема велике хардверске захтеве и погодан је за извршавање у реалном времену. Са друге стране, подлежан је лажно позитивним детекцијама и захтева ручно подешавање параметара. Првенствено детектује фронтална лица и неће радити ако је лице прекривено нечим или није адекватно оријентисано. Треба га користити када је брзина извршавања приоритет и спремни смо да жртвујемо поузданост. HOG алгоритам је поузданији и има стабилнију детекцију. Његови недостаци су што ради само са фронталним (и благо окренутим) лицима јер HOG дескриптор није инваријантан на промене у ротацији и углу гледања, то што није толико поуздан као детектори засновани на дубоком учењу и скуп је у виду захтева за рачунањем јер ради конструкцију клизећег прозора и рачунање HOG фичера за сваки прозор. CNN детектор из Dlib библиотеке је веома поуздан захваљујући самом дизајну алгоритма и добром тренинг скупу, али није погодан за коришћење у реалном времену без GPU убрзања. Може детектовати лица која су под различитим оријентацијама и отпоран је на делимичну прекривеност лица. Не може детектовати лица мања од 80×80 пиксела и оквирни правоугаоник некада може да одсече део чела или браде. Препорука је да се користи када не треба бринути о перформансама у реалном времену. DNN детектор лица из OpenCV библиотеке базиран је на дубоком учењу. У питању је Caffe модел заснован на SSD и ResNet мрежи, брз је и поуздан. Може се извршавати у реалном времену и поузданији је од Viola-Jones и HOG алгоритма, али мање поуздан од CNN из Dlib библиотеке. Модел ради добро када је лице делимично прекривено, када постоје брзи покрети главе и може детектовати бочна лица, али може се десити да не детектује веома мала лица. [39]

За детекцију лица одабран је DNN OpenCV модул. Конкретно, учитана је квантована TensorFlow верзија [40] детектора лица. Поступак детекције лица и обраде детекција је исти као код детекције објеката. Нерегуларним фрејмовима се сматрају фрејмови на којима се не налази тачно једно лице. Овај детектор има еквивалентне функције као детектор особа, с тим што је имплементирана и функција за ресетовање која се позива када детектор не добије фрејм. Ова функција проверава да ли је до тог тренутка било најмање 15 (половина величине прозора) узастопних невалидних фрејмова и ако јесте означава их као нерегуларне. Ако то није случај, проверава да ли је обрађено више од две трећине прекинутог прозора. Ако јесте, уколико је више од половине тих фрејмова нерегуларно, они се пребацују у бафер нерегуларних фрејмова. Слично као код детектора објеката, када овај детектор утврди да на слици не постоји тачно једно лице, ресетоваће се детектори који су хијерархијски испод.

Након детекције лица, детектују се карактеристичке тачке лица које су потребне за даљу обраду лица. Постоји више детектора за ову намену, а један од најпознатијих и најкоришћенијих је детектор који је укључен у Dlib библиотеку. Dlib нуди детектор 68 тачака и детектор који проналази 5 значајних тачака и који је знатно бржи, али препорука је да се он користи ако су нам само потребне локације носа и очију. OpenCV библиотека такође нуди уграђени детектор значајних тачака лица, a MediaPipe библиотека нуди детектор који може детектује 3D мрежу лица. Детекција карактеристичних тачака лица на лицу у овом пројекту обављена је коришћењем модела који детектује 68 тачака из Dlib библиотеке [41]. Детектују се координате тачака које се мапирају на структуру лица и оне су приказане на слици 30. Имплементиране су и помоћне функције које враћају специфичне тачке лица: тачке левог и десног ока (на слици означене бројевима 37-40 и 43-48), горње и доње усне (на слици означене бројевима 49-68) и 6 карактеристичних тачака за естимацију позе главе (на слици означене бројевима 9, 31, 37, 46, 49 и 55).

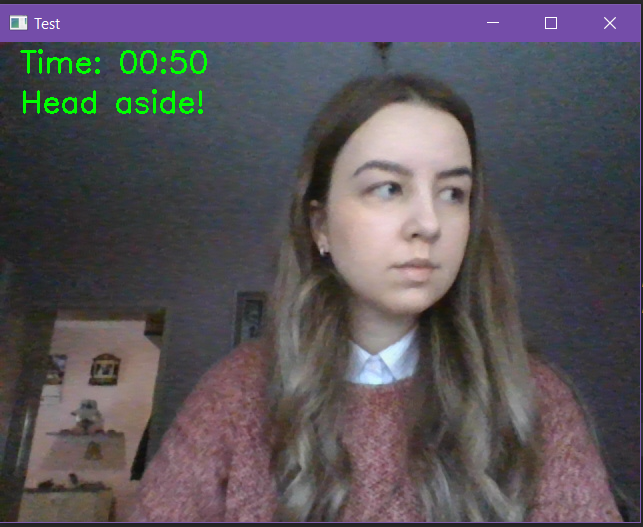
Слика 30: 68 карактеритичних тачака [42] (лево) и детектовано лице са карактеристичним тачкама (десно)

У овом модулу имплементирана је и функција за поравнање лица. Поравнање лица се ради како би се побољшали алгоритми који ће се примењивати над лицем. У нашем случају то су препознавање лица, праћење погледа и трептаја и детекција говора. Желимо да лице буде центрирано на слици, заротирано тако да очи буду на хоризонталној линији и да буде скалирано тако да величина свих лица буде приближно идентична. Све слике су смањене на величину од 256×256 пиксела. Слика са поравнатим лицем се враћа позивајућој функцији и даље провере се врше над овом сликом. За постизање овог ефекта коришћена је афина трансформација. Афине трансформације се користе за ротацију, скалирање и транслацију. Све три трансформације могу се објединити у један позив warpAffine функције из OpenCV библиотеке. За одређивање положаја очију коришћене су претходно издвојене карактеристичне тачке лица. На слици 31 у левом делу приказана је слика где очи особе нису у хоризонтали, а у десном делу обрађена слика где је лице центрирано, увеличано и поравнато.

На почетку се задаје жељена (x, y) позиција левог ока. Координате су изражене у процентима који су најчешће у опсегу 20-40% (у овом пројекту узето је 40%). Ови проценти контролишу колико ће лице бити видљиво након поравнања и варирају од апликације до апликације. Што је мањи проценат лице ће више бити зумирано, а што је већи одзумирано. На основу карактеристичних тачака очију израчунавају се центар једног и другог ока, а након тога и угао који линија која их спаја заклапа са хоризонталом. Овај угао је кључан за поравнање слике. Након тога израчунава се жељена x позиција десног ока (од 1 се одузме жељена x позиција левог ока због симетрије). Сада можемо израчунати жељену дистанцу између два ока по хоризонтали и помножити је са жељеном ширином коначне слике јер је добијена вредност из опсега [0, 1]. Дељењем жељене дистанце са тренутном дистанцом по хоризонтали можемо израчунати фактор скалирања. Затим је потребно наћи централну тачку која се налази између два ока јер ће се око ње вршити ротација. За рачунање ротационе матрице М коришћена је функција cv2.getRotationMatrix2D којој су као параметри прослеђени претходно израчунати центар ротације, угао и фактор скалирања. Добијену матрицу потребно је ажурирати, тачније њену транслациону компоненту како би лице било у оквиру слике и након афине трансформације. Прерачунате су две променљиве: једна у којој је смештена половина жељене ширин слике, и друга у којој се налази висина слике помножена жељеном y координатом левог ока. Транслациона компонента матрице је ажурирана тако што су од ове две вредности одузете одговарајуће координате централне тачке. Након овога се може применити афина трансформација коришћењем функције cv2.warpAffine којој се као параметри прослеђују слика, матрица М, жељена ширина и висина излазне слике и опциони параматар којим се специфицира интерполациони алгоритам. [43]

Слика 31: Оригинални фрејм (лево) и поравнато лице (десно)

Следећа компонента је имплементирана у оквиру модула HeadPoseDetectorПраћење и намењена је детекцији окретања главе. Одређивање положаја главе ученика је имплементирано коришћењем карактеристичних тачака лица и неколико функција из OpenCV библиотеке. Потребно је да знамо да ли је ученикова глава окренута сувише лево, десно, горе или доле. Обрада фрејмова је и у овом случају аналогна обради фрејмова у детекцтору објеката, а ресетовање овог детектора се обавља на исти начин као код детектора лица. Детектор положаја главе се ресетује уколико у фрејму нема детектованих особа и лица. Фрејмови у којима ученик не гледа право у екран се не пропуштају наредним детекторима, већ узрокују њихово ресетовање.

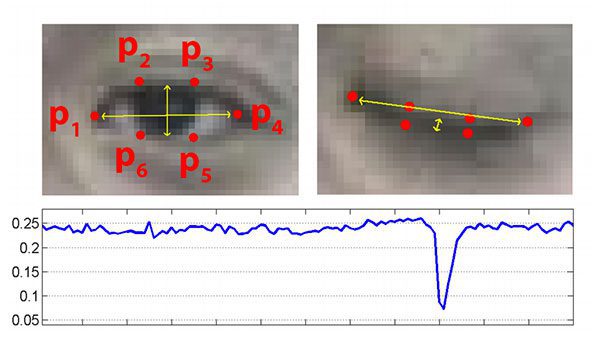
У претходном поглављу описан је PNP проблем. Функција која је коришћена за решавање овог проблема је solvePnP из OpenCV библиотеке. Функција као улазне параметре има низ 3D тачака објекта (у светском координатама), низ одговарајућих тачака на слици (2D тачке лица), унутрашњу (intrinsic) матрицу камере, низ коефицијената дисторзије и метод за решавање PnP проблема (може бити селектовано неколико алгоритама, а подразумевани је SOLVEPNP\_ITERATIVE, који одговара DLT решењу праћеном Levenberg-Marquardt оптимизацијом). **Функција враћа ротациони и транслациони вектор који трансформишу** 3D тачке објекта у координате камере. Затим треба издвојити информације о угловима ротације око координатних оса, односно Ојлерове углове коришћењем RQDecomp3x3 функције. Ова функција захтева ротациону матрицу, а не ротациони вектор, чију конверзију нам омогућава функција Rodrigues. На слици 32 је приказана нерегуларна ситуација где је угао ротације око у-осе већи од граничног. Као координатни почетак се користи врх носа, х-оса је усмерена на лево (из перспективе особе на слици), у-оса на горе, а z-osa у смеру погледа особе. Ротација лево-десно одговара ротацији око у-осе, а ротација горе-доле ротацији око х-осе.

Слика 32: Детекција окретања главе на лево

У оквиру модула LivenessDetector коришћена је детекција трептања ученика за проверу да ли је лице „живо“, односно, да ли није подметнута слика лица. Детектовани су и бројани трептаји ока у одређеном временском интервалу. Ово је један од једноставнијих начина да се изврши ова провера и може се преварити подметањем видео снимка. Особа може припремити снимак себе и подметнути га као виртуелну камеру уместо регуларне камере. Систем би овакав начин преваре могао да детектује тако што ће повремено захтевати од особе која полаже тест да направи одређени покрет, који подметнути снимак неће моћи да прикаже. У сигурносним системима се користе термовизијске (инфрацрвене) камере које региструју емитовање топлоте, односно инфрацрвено зрачење које емитују сва тела, али у условима у којима се треба извршавати овај систем не може се очекивати да ученици поседују овакве камере.

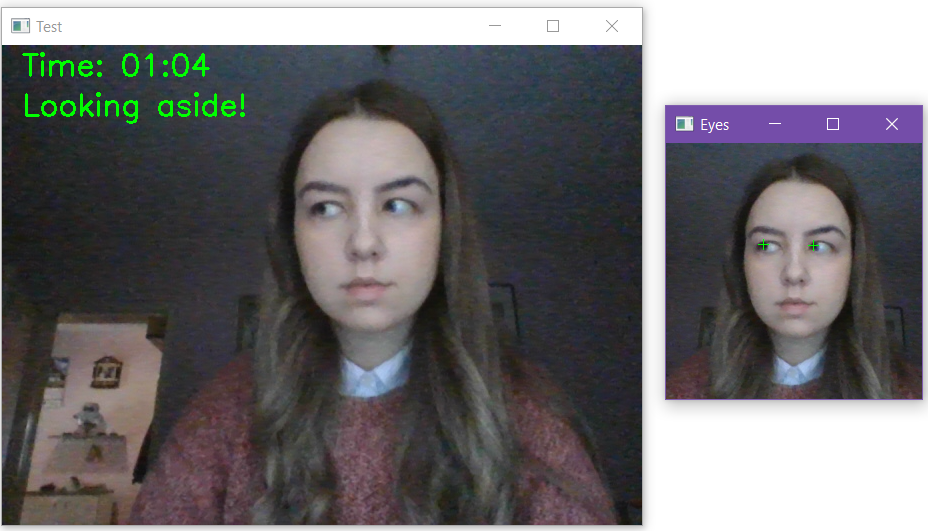
Како би се детектовао трептај коришћена је метрика eye aspect ratio (EAR). За разлику од традиционалних метода за детекцију трептаја које укључују локализацију ока, *thresholding* да би се издвојила беоњача и одређивање да ли бела површина нестаје на одређени временски период, EAR је елегантније решење које врши једноставну калкулацију базирану на односу растојања између одговарајућих карактеристичних тачака очију. Свако око представљено је помоћу 6 тачака, почевши од левог угла ока и крећући се у смеру казаљке на сату. Постоји релација између ширине и висине ових координата:

где су , …,  тачке које одређују око. Ова величина је приближно константна док је око отворено, а нагло опада до нуле када се деси трептај, што се може уочити на графику са слике 33. Коришћењем ове методe избегавамо технике процесирања слике и ослањамо се на растојања између одговарајућих тачака. На графику се може видети EAR кроз време на видео снимку. Вредност је константна, затим нагло опада до нуле, а онда поново расте, што индицира да се десио трептај. [44]

Трептај ока ће трајати неколико фрејмова. Три узастопна фрејма која имају EAR мањи од граничне вредности (0.25) која је експериментално утврђена сматрана су трептајем. Људи у просеку трепну између 12 и 22 пута у току једног минута. Како је прозор од једног минута сувише дуг јер неки тестови могу кратко трајати па можда систем не стигне да детектује неправилност, посматран је прозор од 30 секунди јер се претпоставља да ниједан тест неће трајати краће од тога. Очекује се да ће у том периоду особа трепнути између 6 и 11 пута, али посматран је опсег мало шири од овог због потенцијалних грешака при детекцији. Прозори који имају бројач трептаја ван овог опсега сматрани су нерегуларним. На крају сваког нерегуларног прозора ти фрејмови се пребацују бафер свих нерегуларних фрејмова овог детектора. Услед неправилности у неком од детектора изнад, прозор и бројачи се ресетују.

Слика 33: Детекција трептаја [44]

Ученик би током полагања теста требало да гледа право у екран и да не скреће поглед лево и десно. У модулу EyesTracker детектују се периоди када постоји дуже гледање у једну или другу страну или када се често на кратко гледа са стране, а појединачно краткотрајно гледање са стране се толерише јер то могу бити случајне детекције које не треба санкционисати. И овај детектор садржи бафер који представља прозор фрејмова и функционише на сличан начин као претходно описани детектори. Како се на крају прозора тестира да ли је број фрејмова у којима ученик гледа са стране већи од трећине укупног прозора, биће детектовано ако је за време трајања прозора ученик често гледао са стране на кратко и то ће бити пријављено као проблематична ситуација. Оба ока су праћена истовремено и мерено је растојање зенице од центалног положаја по хоризонтали, чиме се детектује гледање у лево или десно. Поглед на горе или доле није детектован јер су по вертикали знатно мања растојања и незгодно је прецизно мерити их. Детектору се прослеђује поравната слика и карактеристичне тачке једног и другог ока. Прати се центар зенице и рачуна се однос између ширине ока и растојања између левог угла ока и центра зенице. Овај однос је опсегу од 0 до 1, а док особа гледа право вредност је 0.5. Ако је однос мањи од 0,35 сматра се да особа гледа лево, а ако је већи од 0,65 десно. Овај детектор користи извојене карактеристичне тачке ока како би изоловао са слике само регион од интереса. Први корак је издвајање ока, затим тражење одговарајућег прага како би се издвојила зеница и након тога налажење положаја зенице који се користи за одређивање смера гледања.

Око се са слике издваја тако што се направи једна црна слика истих димензија као фрејм, затим слика која служи као маска (бела слика на којој се црним пикселима попуни полигон који представља око) и након тога се примени функција *bitwise\_not* која као параметре има црну слику (улазна), фрејм (излазна) и маску која специфицира који део излазне слике ће бити модификован инвертовањем улазне слике. Добија се бела слика на којој је само око у боји. Након тога је исечен само део те слике на коме је око. Затим се проналази праг који ће најбоље издвојити зеницу ока. Покушава се са вредностима између 5 и 100, са кораком 5, издваја се зеница ока и упоређује се проценат који она заузима у оку са просечним процентом зенице од 0,48. Зеница се издваја тако што се примени функција bilateralFilter над издвојеним оком како би се уклонио нежељени шум, затим се направи кернел 3×3 испуњен јединицама и примени операција ерозије и ако је вредност пиксела мања од threshold-а који се тестира, поставља се на 0, а у супротном на 1, тако да ће бити враћено око које је беле боје, са зеницом црне боје. Проценат који заузима зеница се рачуна као проценат црних пиксела на слици ока. Након проналажења најбољег прага користе се функције *findContours* и *moments* како би се нашле координате центра зенице. [45]

Слика 34: Детекција гледања са стране

Детекција причања је реализована у оквиру модула SpeechDetector праћењем покрета усана. Коришћене су карактеристичне тачке горње и доње усне. Детектор је иницијализован сликом ученика из базе података на којој су ученику уста затворена. Мерена су растојања одговарајућих тачака горње и доње усне. Вредности су унете у 2 низа (један за спољашње, један за унутрашње). Ови низови су упоређивани са истим низовима на улазној слици (фрејму) и ако је разлика већа од унапред задате вредности (експериментално је одабрана вредност 0.2) уста се сматрају отвореним. Креирани су бафери као код претходних детектора. У оквиру прозора фрејмова броје се фрејмови на којима су уста ученика отворена и уколико је при преласку на наредни прозор тај број већи од четвртине обрађених фрејмова пријављује се да је ученик причао и тај прозор се премешта у бафер нерегуларних фрејмова намењен овом детектору. Како се може десити да се ученик прозева или из неког разлога држи уста отвореним дуже време, те ситуације се толеришу и не сматрају невалидним. Постоји бафер који чува узастопне фрејмове у којима су уста отворена. Уколико на преласку између прозора овај бафер има елемената, отварање новог прозора се одлаже док су уста отворена и ти фрејмови се сматрају валидним. Експериментално је утврђено да приликом причања неће бити више од 4 узастопних фрејмова када су уста отворена, тако да се низови дужи од 4 сматрају валидним „зевањем“ и уколико се налазе у оквиру прозора (не на граници) неће се урачунавати у бројање. При ресетовању услед прекида прозора проузрокованог неким од детектора изнад, испитује се да ли је обрађено више од две трећине фрејмова и ако јесте, проверава се да ли је бројач фрејмова са отвореним устима већи од половине обрађених фрејмова, па ако јесте, прозор се сматра нерегуларним. Детекција би била поузданија ако би се користили и подаци са микрофона и детектовали интервали у којима има и гласовне активности и отварања усана. Ово би захтевало креирање две нити, при чему би једна прихватала улаз са камере, а једна улаз са микрофона.

На крају, у сврху препознавања лица у оквиру модула FaceRecognizer коришћен је истрениран модел дубоке неуронске мреже, који је коришћен и у библиотеци face\_recognition [46]. Овај модел има accuracy од 99.38% на *Labeled Faces in the Wild* скупу слика. Мрежа генерише 128 јединствених реалних вредности за сваку улазну слику. Као прозор фрејмова коришћен је бафер величине 50 и у оквиру сваког прозора по једном је вршено препознавање с обзиром на то да је ово захтевнија операција и проузроковала би знатно слабије перформансе када би била извршавана над сваким фрејмом. На почетку је детектору прослеђена поравната слика које је учитана из базе и израчунати су encodings за њу позивом функције *compute\_face\_descriptor* којој се као параметри дају слика лица и карактеристичне тачке лица. Лице са камере упоређивано је са овом сликом, тачније упоређивани су њихови дескриптори и уколико је разлика већа од 0.5, сматра се да лице није препознато, и тај прозор се пребацује се у бафер нерегуларних фрејмова овог детектора. Разлика је рачуната као еуклидско растојање одговарајучих вектора. Нема потребе упоређивати улазну слику са свим сликама из базе. При ресетовању овог детектора, уколико се до момента ресета детектовао невалидни фрејм, фрејмови из тог прозора који су до тада обрађени означавају се као нерегуларни.

# **Закључак**

Софтвер за полагање онлајн тестова је све напреднији и све више се користи за потребе различитих тестирања. Предност оваквог начина тестирања су смањене потешкоће око организације јер нема потребе за проналажењем одговараће просторије и особа које би дежурале, штеди се на времену људи и брже се генеришу и анализирају резултати теста. Вештачка интелигенција и машинско учење значајно доприносе реализацији тестирања на даљину јер софтвери за аутоматско надгледање полагања у доброј мери омогућавају детекцију неправилности.

Приликом тестирања могу се надгледати микрофон, камера и екран особе која полаже тест и најпрецизнија анализа добија се комбинацијом сва три извора. У овом раду дат је преглед техника рачунарског вида које се могу приметити у обради података само са камере и реализован је прототип система који примењује неке од њих. Обрађени су ранији алгоритми као што су Viola-Jones, HOG и Eigenfaces, а и модернији методи засновани на дубоким неуронским мрежама који су данас најзаступљенији. За детекцију објеката најчешће се користе модели као што су YOLO и SSD. Детекција и препознавање лица такође имају најбоље перформансе када се користе неуронске мреже. За праћење покрета главе, очију и усана потребно је познавати карактеристичне тачке лица и пратити их у сукцесивним фрејмовима.

Прототип система који је реализован је доста једноставнији од система који би могао да се користи у реалности. При одабиру алгоритама узето је у обзир окружење у коме систем треба да се извршава и доступни хардверски ресурси. Камера има увид у ограничен део простора, тако да поред особе која ради тест може седети неко и помагати јој, а да то не буде примећено. Овакви пропусти би могли бити превазиђени уколико би се захтевао и снимак комплетног окружења у коме се полаже тест и уколико би се детектовао и говор јер је најчешће недозвољено понашање на онлајн тестовима тражење вербалне помоћи од неке друге особе.

Увек се могу наћи начини да се превари систем и још увек се не можемо у потпуности ослонити на аутоматско надгледање тестирања и искључити људски фактор, али чињеница је да су ови системи све напреднији.

# **Литература**

1. Why Live Proctoring of Online Exams Is Becoming Mainstream, <https://www.testreach.com/blog-post/proctoring-online-exams.html?fbclid=IwAR0r8C4Cg4BLoVv-VJdbadDFxZk8qAiA_HHEkq7mLrEGkpa-U_7iah8jG10>, [последњи приступ: 01.10.2022.]
2. 12 Best Proctoring Software for Cheating-Free Online Exams, <https://www.techjockey.com/blog/proctoring-software-for-online-exam>, [последњи приступ: 01.10.2022.]
3. Can An Online Exam Detect Cheating?, <https://www.masterteachingonline.com/can-an-online-exam-detect-cheating/?fbclid=IwAR0CTcD3-2ixdQQaBXkrii_eVHSyffVz2AZ4P08q0tc4cSRFCm0-VVLCOPs>, [последњи приступ: 08.10.2022.]
4. Honorlock, <https://honorlock.com/>, [последњи приступ: 06.10.2022.]
5. ProctorEdu, <https://proctoredu.com/>, [последњи приступ: 06.10.2022.]
6. Features of the Exam Platform, <https://pages.mettl.com/faq-online-examination#Cheating-Prevention>, [последњи приступ: 06.10.2022.]
7. <https://examus.com/>, [последњи приступ: 14.11.2022.]
8. Computer vision, <https://en.wikipedia.org/wiki/Computer_vision>, [последњи приступ: 23.10.2022.]
9. Object detection, <https://en.wikipedia.org/wiki/Object_detection>, [последњи приступ: 24.10.2022.]
10. Object Detection in 2022: The Definitive Guide, <https://viso.ai/deep-learning/object-detection/>, [последњи приступ: 24.10.2022.]
11. P. Viola, M. Jones, Rapid Object Detection using a Boosted Cascade of Simple Features, <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf>, [последњи приступ: 26.10.2022.]
12. N. Dalal, B. Triggs, Histograms of Oriented Gradients for Human Detection, <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>, [последњи приступ: 28.10.2022.]
13. Histogram of Oriented Gradients explained using OpenCV, <https://learnopencv.com/histogram-of-oriented-gradients/>, [последњи приступ: 31.10.2022.]
14. Support Vector Machine — Introduction to Machine Learning Algorithms, <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>, [последњи приступ: 31.10.2022.]
15. R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, <https://arxiv.org/pdf/1311.2524.pdf>, [последњи приступ: 01.11.2022.]
16. M. Dabović, I. Tartalja, Duboke konvolucijske neuronske mreže – koncepti i aktuelna istraživanja, <https://www.researchgate.net/publication/321709294_Duboke_konvolucijske_neuronske_mreze_-_koncepti_i_aktuelna_istrazivanja>, [последњи приступ: 01.11.2022.]
17. R. Girshick, Fast R-CNN, <https://arxiv.org/pdf/1504.08083.pdf>, [последњи приступ: 04.11.2022.]
18. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, <https://arxiv.org/pdf/1506.01497.pdf>, [последњи приступ: 04.11.2022.]
19. K. He, G. Gkioxarim, P. Dollár, R. Girshick, Mask R-CNN, <https://arxiv.org/pdf/1703.06870.pdf>, [последњи приступ: 04.11.2022.]
20. A. Pramanik, S. K. Pal, J. Maiti, P. Mitra, Granulated RCNN and Multi-Class Deep SORT for Multi-Object Detection and Tracking, <https://www.researchgate.net/publication/348254614_Granulated_RCNN_and_Multi-Class_Deep_SORT_for_Multi-Object_Detection_and_Tracking>, [последњи приступ: 08.11.2022.]
21. J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, <https://arxiv.org/pdf/1506.02640.pdf>, [последњи приступ: 09.11.2022.]
22. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. C. Berg, SSD: Single Shot MultiBox Detector, <https://arxiv.org/pdf/1512.02325.pdf>, [последњи приступ: 10.11.2022.]
23. SSD object detection: Single Shot MultiBox Detector for real-time processing, <https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06>, [последњи приступ: 10.11.2022.]
24. Face detection, <https://en.wikipedia.org/wiki/Face_detection>, [последњи приступ: 10.11.2022.]
25. Facial recognition system, <https://en.wikipedia.org/wiki/Facial_recognition_system>, [последњи приступ: 10.11.2022.]
26. Liveness Detection with OpenCV, <https://pyimagesearch.com/2019/03/11/liveness-detection-with-opencv/>, [последњи приступ: 09.11.2022.]
27. What is face recognition?, <https://pyimagesearch.com/2021/05/01/what-is-face-recognition/>, [последњи приступ: 09.11.2022.]
28. M. A. Turk, A. P. Pentland, Face Recognition Using Eigenfaces, <https://sites.cs.ucsb.edu/~mturk/Papers/mturk-CVPR91.pdf>, [последњи приступ: 10.11.2022.]
29. ML | Face Recognition Using Eigenfaces (PCA Algorithm), <https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/>, [последњи приступ: 10.11.2022.]
30. P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection <https://web.ece.ucsb.edu/~hespanha/published/faces.pdf>, [последњи приступ: 10.11.2022.]
31. T. Ahonen, A. Hadid, M. Pietikäinen, Face Recognition with Local Binary Patterns, <https://link.springer.com/content/pdf/10.1007/978-3-540-24670-1_36.pdf>, [последњи приступ: 10.11.2022.]
32. Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning, <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>, [последњи приступ: 10.11.2022.]
33. Head Pose Estimation using OpenCV and Dlib, <https://learnopencv.com/head-pose-estimation-using-opencv-and-dlib>, [последњи приступ: 11.11.2022.]
34. Gaze Tracking, <https://learnopencv.com/gaze-tracking/>, [последњи приступ: 11.11.2022.]
35. Voice Activity Detection, [https://maelfabien.github.io/machinelearning/Speech4/?fbclid=IwAR2V3yKH5REzQ\_LgUDfCkN1X52fkQ10io7H\_4Q-iKCrtOcjyhupWxlB4wwQ#](https://maelfabien.github.io/machinelearning/Speech4/?fbclid=IwAR2V3yKH5REzQ_LgUDfCkN1X52fkQ10io7H_4Q-iKCrtOcjyhupWxlB4wwQ), [последњи приступ: 12.11.2022.]
36. Speech recognition, <https://en.wikipedia.org/wiki/Speech_recognition>, [последњи приступ: 12.11.2022.]
37. TensorFlow Object Detection API, <https://github.com/opencv/opencv/wiki/TensorFlow-Object-Detection-API>, [последњи приступ: 28.10.2022.]
38. COCO, <https://cocodataset.org/#home>, [последњи приступ: 28.10.2022.]
39. Face detection tips, suggestions, and best practices, <https://pyimagesearch.com/2021/04/26/face-detection-tips-suggestions-and-best-practices/>, [последњи приступ: 04.10.2022.]
40. <https://github.com/opencv/opencv/tree/4.x/samples/dnn/face_detector>, [последњи приступ: 04.10.2022.]
41. <http://dlib.net/face_landmark_detection.py.html>, [последњи приступ: 01.10.2022.]
42. Facial point annotations, <https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/>, [последњи приступ: 01.10.2022.]
43. Face Alignment with OpenCV and Python, <https://pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/>, [последњи приступ: 13.11.2022.]
44. T. Soukupová, J. Čech, Real-Time Eye Blink Detection using Facial Landmarks, <http://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf>, [последњи приступ: 02.11.2022.]
45. Gaze Tracking, <https://github.com/antoinelame/GazeTracking>, [последњи приступ: 14.11.2022.]
46. Face Recognition, <https://github.com/ageitgey/face_recognition#face-recognition>, [последњи приступ: 14.11.2022.]