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](#_Toc115642908)

[**Софтвери за праћење регуларности полагања тестова** 4](#_Toc115642909)

[Аутоматско надгледање онлајн тестирања 4](#_Toc115642910)

[Примери софтвера за аутоматско надгледање онлајн тестирања 5](#_Toc115642911)

[**Технике рачунарског вида за праћење регуларности полагања тестова** 8](#_Toc115642912)

[Детекција објеката 8](#_Toc115642913)

Детекција и [препознавање лица 8](#_Toc115642913)

[Одређивање положаја главе 8](#_Toc115642913)

[Праћење покрета очију и усана 8](#_Toc115642913)

[**Имплементација система за праћење регуларности полагања тестова** 37](#_Toc115642924)

[Детекција објеката на камери 8](#_Toc115642913)

[Детекција лица и карактеристичних тачака лица 8](#_Toc115642913)

[Детекција окретања главе 8](#_Toc115642913)

[Провера да ли је лице реално 8](#_Toc115642913)

[Праћење погледа 8](#_Toc115642913)

[Детекција говора 8](#_Toc115642913)

[Препознавање лица 8](#_Toc115642913)

[**Закључак** 8](#_Toc115642912)

[**Литература** 8](#_Toc115642912)

# **Увод**

Пандемија настала услед ширења вируса Ковид-19 утицала је на све аспекте живота људи, па тако и на образовање. У условима који су настали као последица ове ситуације било је неопходно прилагодити начин извођења наставе како би се школска година одвијала несметано. Многе образовне институције биле су приморане да пређу на хибридни или у потпуности онлајн модел одржавања наставе, те је један од проблема био како пронаћи најбољи начин за оцењивање ученика/студената.

Полагање онлајн тестова на рачунарима и сличним уређајима 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 већина наставника одабрала овакве тестове као меру оцене. Наравно, тиме се отвара простор за различите начине преписивања. Неки од начина да се преписивање спречи су увођење временског ограничења за одговор за свако питање, онемогућавање враћања на претходно одговорено питање, ограничење да сви морају полагати тест истовремено, генерисање различитих питања, приказ питања и понуђених одговора у различитом редоследу... Међутим, ове мере некада нису довољне и ученици/студенти ипак пронађу начин да потраже туђу помоћ. Из тог разлога направљен је софтвер за надгледање полагања онлајн тестирања и генерисање одговарајућег извештаја.

Једано од решења је да дежурна особа надгледа ученике током полагања у реалном времену. Могу се пратити микрофон, камера и екран ученика који полаже тест. Друго решење подразумева снимање ученика и његовог екрана током полагања и касније прегледање тог снимка ради контроле регуларности. Предност оваквог решења је што ученик и дежурни не морају у заказано време бити присутни, већ касније може бити прегледана убрзана верзија комплетног снимка полагања. У овом случају нема могућности да дежурна особа реагује на неправилности у реалном времену, због којих је можда требало онемогућити наставак полагања теста ученику. Треће решење је напредно, аутоматско нагледање полагања, где вештачка интелигенција има кључну улогу у препознавању нерегуларности. Коришћење алгоритама вештачке интелигенције и машинског учења у великој мери мења особу која је била задужена да лично надгледа снимке свих ученика који полажу тестове. Систем генерише упозорења током полагања и бележи све нерегуларне ситуације које су детектоване и које се касније поново прегледају како би се потврдиле неправилности. Још увек не можемо у потпуности да се ослонимо на рачунар и искључимо људски фактор у надгледању регуларности, али овакви системи се могу користити у реалном времену и у будућности ће можда бити довољно поуздани да могу самостално да обављају овај посао. [1]

У овом раду биће обрађене методе рачунарског вида које се могу користити за обраду и валидацију 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 овај проблем. У трећем поглављу обрађене су технике и алгоритми рачунарског вида који се могу применити за детекцију особа и лица на видео снимку, препознавање лица, праћење положаја главе и погледа и детекцију говора ученика који полаже тест. У наредном поглављу описана је конкретна имплементација прототипа система који се може користити за надгледање регуларности полагања тестова на основу података са камере. Систем детектује невалидне ситуације, 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 пројекти који су развијени у циљу омогућавања реализације учења и тестирања на даљину. Углавном се ради о веб апликацијама, а има и апликација које је потребно инсталирати на свом рачунару или мобилном телефону. Ови софтвери, поред примене у образовним институцијама, могу се користити и у различитим компанијама и организацијама за потребе било ког типа тестирања попут интервјуа за посао, семинара и добијања сертификата.

Као најважнији захтеви које је потребно да ови софтвери испуне могу се издвојити: једноставност коришћења, превенција варања, сигурност и приватност, скалабилност, прилагодљивост и интеграција са другим сервисима. Једноставност коришћења је веома важна јер софтвер треба да корисницима пружи добро корисничко искуство и не захтева превише њиховог времена. Алати и технике за детекцију нерагуларности су неопходни како би тестирање било еквивалентно тестирању у заједничкој просторији у којој једна или више особа дежура. 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. Такође, систем треба да буде једноставан за интеграцију са постојећим платформама за учење попут LMS-а[[1]](#footnote-1) (енг. Learning Management System).[2]

Као што је напоменуто у уводном поглављу, софтвер за надгледање онлајн тестирања може се поделити у три групе. У прву групу спадају софтвери код којих особа која је задужена за надгледање регуларности уживо посматра ученика који полаже тест и може одмах реаговати уколико примети неке неправилности. Систем захтева од ученика који полаже тест да укључи камеру, микрофон и подели свој екран како би дежурна особа имала увид у све његове активности. Други начин за реализацију система је да се током полагања ученик и његов екран снимају, а касније прегледа убрзана верзија тог снимка ради утврђивања евентуалних нерегуларности. Софтвери из треће групе не захтевају ангажовање дежурне особе током тестирања, засновани су на техникама вештачке интелигенције и рачунарског вида и све више се користе у пракси. У овом поглављу биће дат преглед метода за аутоматско надгледање онлајн тестирања и неколико најпопуларнијих производа који су доступни и користе их многобројне институције и организације.

## Аутоматско надгледање онлајн тестирања

Софтвери за онлајн учење и тестирање углавном имају механизам за регистровање корисника и захтевају да се ученици улогују да би могли да користе апликацију. Ово је први корак у процесу полагања и може бити имплементиран на различите начине, али најчешће се ученик пријављује на систем коришћењем свог и-мејла, а пре самог покретања теста мора да коришћењем камере покаже своју идентификациону картицу или омогући систему да забележи његову слику и упореди 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. Испитује се да ли ученик разговара са неким, да ли је учеников глас исти током трајања теста и у неким случајевима се може поредити са гласом тог ученика из базе података. Надгледање звука може спречити да неко други ради тест уместо ученика, да му нека особа помаже „добацивајући“ му одговоре или да му одређена апликација саопштава одговоре на питања. Модул који је задужен за праћење екрана кандидата углавном санкционише сликање и снимање прозора у ком је отворен тест, напуштање теста и отварање других страница, апликација или фајлова. Софтвер има контролу над веб претраживачем рачунара у коме ученик полаже тест. Забрањује се отварање апликација које могу олакшати дељење одговора нудећи могућност за дописивање и дељење екрана. Софтвери који захтевају да камера ученика буде укључена током полагања теста могу надгледати његове покрете и пријавити сумњиво понашање. Модул за обраду података са камере може да контролише да ли у кадру има других особа или недозвољених уређаја попут мобилних телефона, да врши препознавање лица ученика, могу се пратити покрети очију ученика и детектовати када он дуже време гледа са стране, или када често на краћи временски период гледа са стране. Софтвер за надгледање 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]

## Примери софтвера за аутоматско надгледање онлајн тестирања

На тржишту су доступни различити софтвери за тестирање. За креирање кратких тестова који не захтевају висок степен сигурности и провере, најједноставније је користити неки од бесплатних веб сајтова за генерисање квизова и тестова. Када су у питању озбиљнији тестови који захтевају висок степен сигурности и надгледања, треба одабрати неки од напреднијих софтвера, у зависности од потреба и буџета организације.

Једно од водећих решења је Honorlock, први сервис за надгледање онлајн тестирањања који комбинује аутоматско надгледање са људским прокторима у циљу очувања интегритета теста. Тестирање надгледа вештачка интелигенција и ако детектује академску нечеститост алармира проктора да се прикључи сесији у реалном времену. Овиме се губи потреба да нека особа све време надгледа више кандидата, што и кандидатима смањује осећај нелагодности због присуства друге особе. Циљ није само детектовати варање, већ учинити сам процес тестирања што лагоднијим како за оног ко полаже тако и за особу која дежура. Апликација нуди интерфејс који је веома једноставан за коришћење, доступна је 24/7, сигурна и скалабилна. 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, 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, претраживачима и апликацијама, при чему закључава и пречице на тестатури попут штампања, копирања и лепљења (енг. copy/paste), сликања екрана, 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. Од ученика се може затражити да преузме додатну апликацију која је задужена за праћење других апликација које ученик отвара током полагања теста на свом уређају. Могућа је директна интеграција са LMS платформама попут MyMathLab, ALEKS, Pearson, и McGraw Hill. 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. Honorlock 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. Недостацима се може сматрати ограничење да институција мора имати најмање 250 корисника како би могла да користи апликацију, није доступна бесплатна пробна (енг. trial) верзија и тренутно ради само са Google Chrome претраживачем, односно постоји екстензија за овај претраживач. [4]

ProctorEdu је веб претраживач-решење за онлајн надгледање, снимање и евалуацију корисничног понашања током онлајн тестирања, без потребе да корисник преузима на свом уређају било какав софтвер. Компанија се бави овом темом од 2016. године и фокусирана је на корисничко искуство. Функционалност надгледања је интегрисана са LMS или платформама за тестирање и омогућава надгледање удаљеног тестирања уживо или у аутоматском моду. Око 2 милиона тестова до сада је одрађено преко ове платгорме укључујући до 5 000 особа које инстовремено раде тест. Софтвер нуди поуздане резултате теста, биометријску аутентификацију, подршку за Android и iOS мобилне уређаје, 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). Приликом организовања теста може се доделити више проктора који ће надгледати полагање, а систем ће им помоћи тако што ће их обавештавати о нерегуларним активностима у реалном времену. Проктори могу комуницирати са ученицима путем чета, видеа или аудија уколико ученици буду имали неких питања. Особа која полаже тест покреће проверу опреме која траје 30 секунди, доказује свој идентитет показивањем идентификационе картице, гледа видео са правилима и покреће тест. Систем нуди функционалност препознавања лица током трајања теста, детекцију буке, контролише покушај претраге на интернету. Извештај да ли је било невалидности се генерише у видео и пдф формату. Подржава различите претраживаче попут Chrome, Opera, Firefox, Edge, Safari. [5]

Mercer **|** Mettl такође нуди безбедно и фер 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, корисничку подршку у свако доба, али не и бесплатну верзију. Може се користити као комплетна платформа за тестирање, али 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 других особа и говор других особа. Студент се пре почетка теста пријављује уз помоћ и-мејл аутентификације, мобилне аутентификације коришћењем OTP (енг. One-time password) или своје студентске идентификационе картице, а током трајања теста се у одређеним временским интервалима скенира слика студента 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. Систем се може скалирати у зависности од потреба. У понуди је и Mettl Secure Browser који искључује све екстерне портове, спречавајући кандидата да повеже секундарни екран. Browser такође искључује све софтвере за дељење садржаја и вебсајтове и не допушта напуштање прозора у коме се ради тест пре предаје теста. AI модул је обучен да детектује до 18 типова нерегуларности. [6]

SpeedExam је још једна платформа за онлајн тестирање која нуди напредне функционалности као што су аутоматско оцењивање и брзи извештаји. Велика предност ове платформе је што нуди бесплатну верзију која није временски ограничена и долази са 25 бесплатних тестова месечно, а цена је генерално најповољнија у односу на конкуренцију. Апликација је сагласна са GDPR (енг. General Data Protection Regulation) и омогућава енкриптовано складиштење података. Овај софтвер користи више од 10 000 користика као што су Microsoft, Amazon, Samsung, University of Toronto, Hewlett Packard Enterprise и многи други. Омогућава 30 000 истовремених тестова, менаџмент полагања и кандидата, као и напредне извештаје и анализу. Када нерегуларне активности пређу одређени ниво, систем онемогућава наставак полагања кандидату. Апликација нуди интерфејс на различитим језицима. [7]

Examus је један од најфлексибилнијих сервиса за онлајн надгледање. Користи веб камеру за препознавање лица и детекцију емоција, праћење погледа, детекцију других особа и гласова. Може се користити на мобилним телефонима, нуди једноставан API, високо је скалабилан, штити корисничке податке. Examus нуди два главна производа: Proctor AI – софтвер за онлајн надгледање коришћењем вештачке интелигенције и Examus EQ – алат који надгледа човекову пажњу и ангажпвање током похађања онлајн курсева. AI proctor 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. Систем детектује. 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, покушај дељења екрана или звука. Како нечија судбина не би зависила искључиво од алгоритама, постоји могућност укључивања људског надгледача који ће у реалном времену проверити забележене нерегуларности. Компанија нуди обучено особље које може кориговати генерисани извештај и проследити неуронској мрежи одговрајуће параметре како би се повећала поузданост извештаја. Са друге стране, Examus EQ, систем за праћење пажње и ангажовања у реалном времену, процесира видео улаз и шаље резулатате анализе клијенту, што омогућава да предавачи добију у реалном времену информацију о томе како публика реагује на њихова предавања и вебинаре. Ово омогућава увођење промена у начину излагања у ходу. Систем може доставити аналитичке податке за комплетну публику или за сваког појединца. Обрада се врши на клијентској страни, што минимизује трошкове израчунавања на серверској страни и избегава потребу за слањем приватних података. Examus EQ је интегрисан са YouTube, што значи да корисник може делити видео који је уплоадован на YouTube са колегама и ученицима и да добије детаљну анализу утисака. Исто важи и за уживо (енг. live) пренос на YouTube. [8]

# **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. [9]

Неке од техника рачунарског вида које се могу применити у апликацијама за праћење регуларности полагања онлајн тестова су детекција објеката, детекција и препознавање лица, одређивање положаја главе и праћење покрета очију и усана особе која полаже тест.

## Детекција објеката

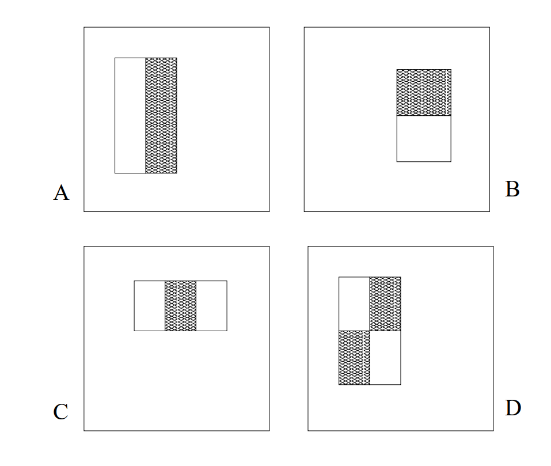
Детекција објеката је рачунарска технологија која спада у област рачунарског вида и обраде слика и бави се проналажењем објеката који припадају одређеним класама (нпр. људи, зграде, аутомобили) на дигиталним сликама и видео снимцима. [10] Свака класа објеката има специфичне особине, такозване фичере (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 неопходно је издвојити фичере објеката коришћењем неког од детектора фичера и након тога користити неку од техника за класификацију, попут SVM (енг. Support Vector Machine). Најпознатији примери ових метода су Viola–Jones детектор и HOG (енг. Histogram of Оriented Gradients) детектор. Технике базиране на дубоком учењу најчешће се ослањају на конволуционе неуронске мреже (CNN, енг. Convolutional Neural Networks). Данас су рачунари много ефикаснији него што је то било раније и тежи се паралелизацији процеса обраде, што омогућава детекцију објеката у приближно реалном времену. Дубоке конволуционе неуронске мреже у комбинацији са убрзањем које доноси графички процесор[[2]](#footnote-2) значајно су унапредиле процес детекције објеката. Детекција објеката захтева обављање два задатка: први је проналажење произвољног броја региона објеката, а други задатак је класификација сваког од њих и одређивање правоугаоника којим се објекат може уоквирити. Поступак се може реализовати кроз две фазе, а може и у оквиру једног корака како би се добиле боље перформансе. Детектори који раде у 2 фазе имају велику поузданост, али су спорији. Најпознатији двофазни детектори су R-CNN, Fast R-CNN, Faster R-CNN и Mask R-CNN. Детектори који раде у једном кораку 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 алгоритама. Најпознатији детектори из ове групе су YOLO (енг. You Only Look Once) који има више верзија и SSD (енг. Single Shot MultiBox Detector). [12]

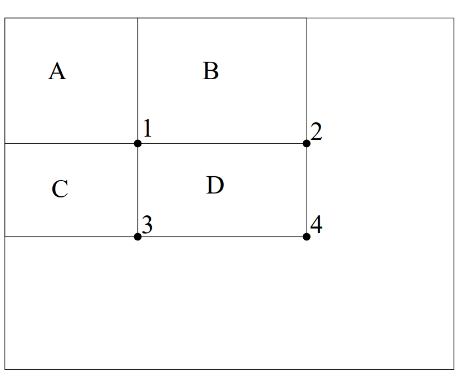
Раније методе могу бити ограничене услед постојања комплексне позадине, делимично прекривених објеката, лошег осветљења и шума, док су технике дубоког учења су значајно отпорније на наведене проблеме, али им је недостатак што захтевају велику количину података за тренирање и што процес означавања слика може бити временски захтеван. 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 верзије R-CNN-a, YOLO и SSD.

Viola-Jones (Haar) детектор

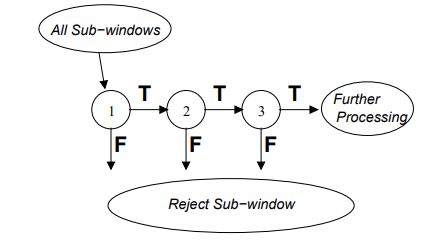
Paul Viola и Michael Jones су у свом раду [13] из 2001. године представили метод за детекцију објеката у реалном времену реализован кроз примену неколико нових техника. Првобитно је био намењен аутоматској детекцији лица, али може се користити за детекцију било ког типа објеката.

За детекцију је коришћен скуп фичера приказаних на слици 1 који подсећају на Haar-ове таласе[[3]](#footnote-3) (одакле и потиче назив детектора). 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.

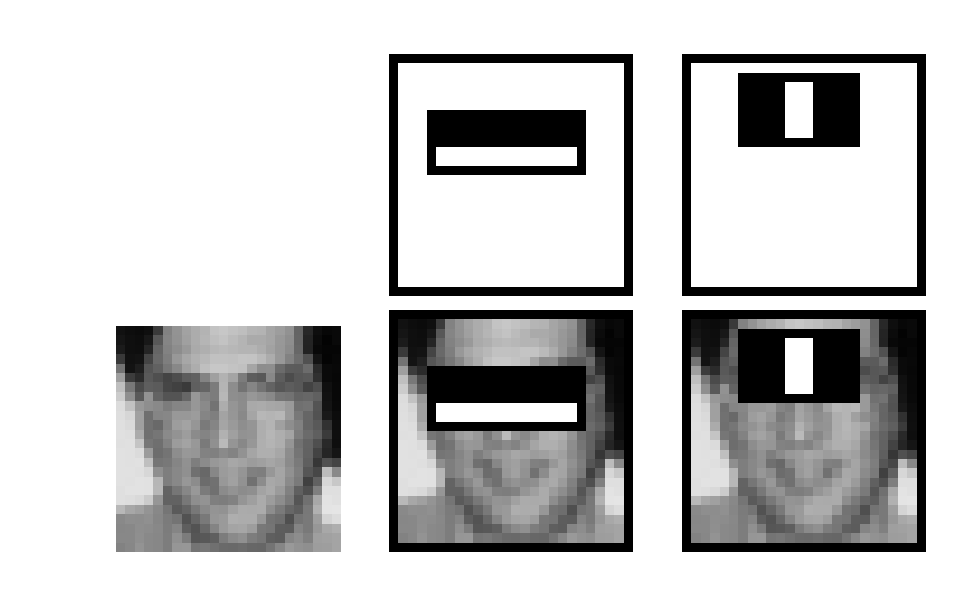
Слика 1: Haar фичери [13]

Аутори су такође приказали нови начин за репрезентацију слике назван „интегрална слика“, која 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 А. Вредност у тачки 2 је , у тачки 3 је , а у тачки 4 је . Сума пиксела у оквиру правоугаоника D би била . 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 [13]

За прозор димензија 24×24 се израчунавају фичери (око 180 000 фичера). Иако се појединачни фичери израчунавају прилично једноставно и брзо, рачунање комплетног скупа фичера је скупо. Да би се издвојио мањи скуп најважнијих фичера из великог скупа фичера 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 један од 180 000 фичера 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 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 (слика 3). 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 [13]

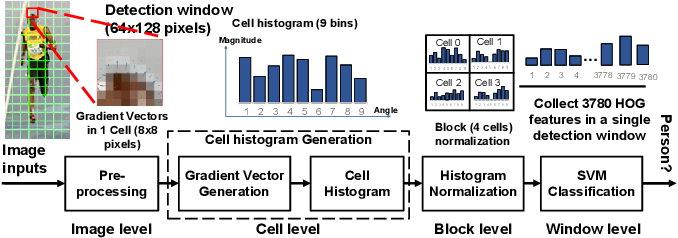
У раду је детекција објеката демонстрирана на примеру детекције лица. На слици 4 приказана су прва два фичера селектована AdaBoost алгоритмом. Први фичер наглашава особину лица да је регион око очију углавном тамнији од региона горњег дела образа, док други фичер осликава особину да је регион очију тамнији 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. Детектор обилази слику која се тестира методом клизајућег прозора у више величина, при чему се детектор (прозор) скалира, а не слика. 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. [13]

Овај алгоритам 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 у реалном времену, а недостатак што је склон лажно позитивним детекцијама и детектује углавном само фронтално постављене објекте. И даље се често користи када се детекција врши на ограниченим уређајима на којима не можемо извршавати детекторе који су рачунски захтевнији.

Слика 4: Прва два фичера селектована AdaBoost техником [13]

HOG детектор

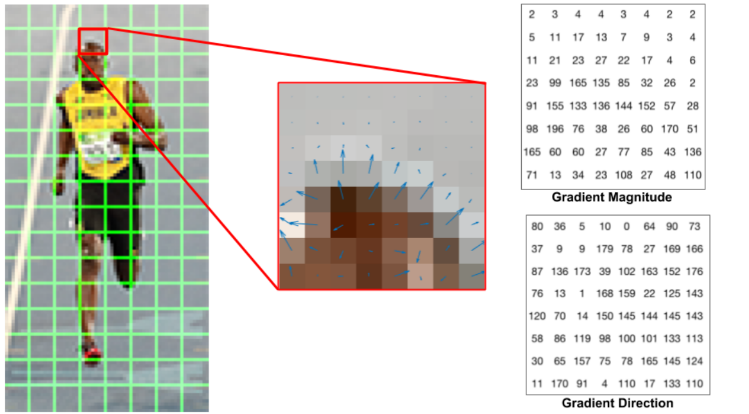
У раду [15] из 2005. године показано је да се за детекцију објеката на слици могу успешно користити HOG фичер дескриптор слике и линеарни SVM класификатор (који је коришћен због своје једноставности и брзине). Аутори су се фокусирали на детекцију особа на слици, али детектор се може применити и за детекцију објеката других класа. Примењује се пирамида слика и метода клизајућег прозора како би се детектовали објекти различитих величина.

Фичер дескриптор је репрезентација слике која издваја корисне информације из ње, а одбацује ирелевантне. Код HOG дескриптора, улазна слика димензија 64 × 128 × 3 (канала) се 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 магнитуда градијената велика око ивица и ћошкова (региони наглих промена интензитета), а они носе много више информација о облику објекта него равни региони.

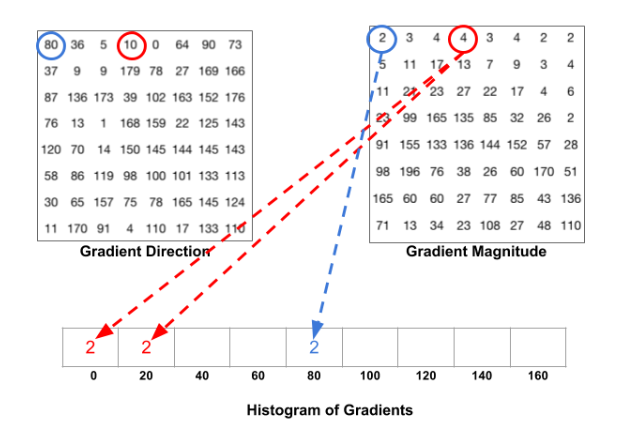
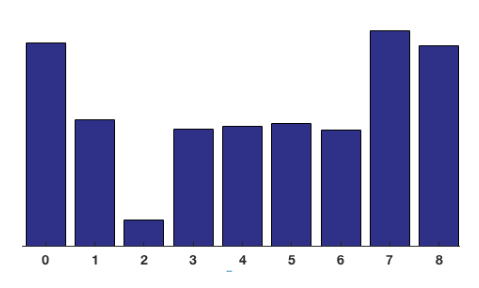
Слика 4: Начин рада HOG детектора [16]

На слици 4 приказан је процес детекције објеката HOGдетектором*.* Први корак у рачунању хистограма оријентисаних градијената је препроцесирање слике. У оригиналном раду HOG дескриптор фичера је рачунат над деловима слике величине 64×128, али у општем случају слика може бити било које величине. Посматрају се делови слике са различитим скалирањем и на различитим локацијама. Једино ограничење је да парче које се обрађује има фиксни aspect ratio (у нашем случају 1:2). Дакле, са слике се издвоји део који има овај aspect ratio и димензије му се промене на 64×128. Аутори су у раду обавили и гама корекцију у оквиру препроцесирања. Затим 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 Gx и Gy градијенти по x и y осама. На слици 5 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 максималном градијенту.

Слика 5: Лево : x-извод слике. Центар: y-извод слике. Десно : Магнитуда извода. [16]

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 парчета садржи 2 вредности (магнитуду и смер) за сваки пиксел, што даје бројева. Индивидуални градијенти могу садржати шум, али хистограм над 8×8 парчетом 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.

Слика 6: Лево: 8×8 ћелије. Центар : парче слике и градијенти представљено стрелицама. Десно : Градијенти истог парчета слике представљени бројевима [16]

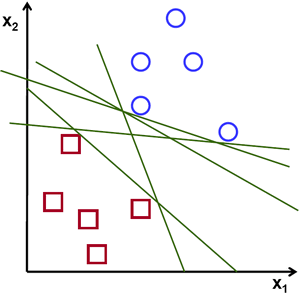
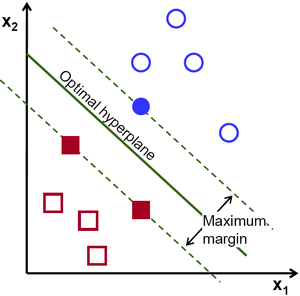
Хистограм је низ од 9 елемената који одговарају угловима 0, 20, 40, 60, …, 160 степени. На слици 6 у средини приказано је парче слике на коме су стрелицама представљени смерови градијената, а дужина стрелица представља магнитуду. Смерови стрелица показују смер промене интензитета, а њихова дужина колико је велика разлика. Десно можемо видети бројеве који репрезентују градијенте у 8×8 ћелијама, уз разлику што су углови између 0 и 180 степени уместо 0 и 360. Ово су „неозначени“ градијенти јер су градијент и његова негативна варијанта представљени истим бројем.

Слика 8: Хистограм [16]

Слика 7: Хистограм градијената [16]

Следећи корак је креирање хистограма градијената у оквиру ових 8×8 ћелија. Хистограм садржи 9 елеменета који одговарају угловима 0, 20, 40, …, 160. Слика 7 илуструје процес. Посматрамо исто парче као на претходној слици. Елемент је селектован на основу смера, а вредност елемента на основу магнитуде. Ако погледамо пиксел означен плавом бојом, приметићемо да он има угао (смер) од 80 степени и магнутуду 2, тако да 2 иде на 5. позицију. Градијент пиксела заокружен црвеном бојом има угао од 10 степени који је на средини између 0 и 20, тако да се његова вредност 4 уписује на оба места (по 2). Ако је угао већи од 160 степени, он је између 160 и 180 степени, а угао 0 и 180 су еквивалентни, тако да би се његова вредност пропорционално уписала у елемент намењен углу од 0 и 160 степени. На слици 8 је приказано како би изгледао хистограм за описани пример. Y-оса представља 0 степени. Како у хистограму доминирају вредности близу 0 тј. 180 степени, може се закључити да ова ћелија вероватно садржи ивицу јер су градијенти углавном усмерени на горе или доле. Како би дескриптор био инваријантан на промене у осветљењу, треба да нормализујемо хистограм. Нормализација подразумева дељење сваке вредности вектора са дужином вектора (L2 нормом). Два вектора која се разликују само у скалирању свешће се на исти вектор након нормализације. Посматрамо блок 16×16, он има 4 хистограма који када се надовежу 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 (36×105, где је број позиција на којима може да се нађе прозор 16×16). Овај вектор се сада користи за тренирање класификатора као што је SVM како би се обавила детекција објеката. [16]

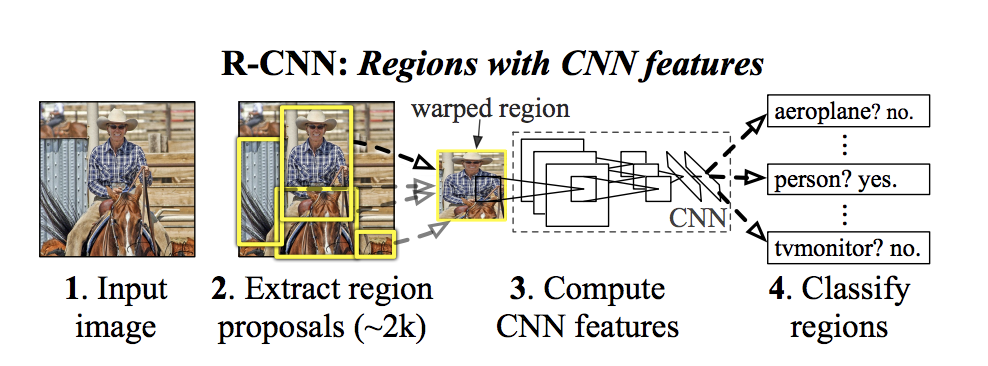
SVM (Support Vector Machine) је техника класификације која користи нелинеарно мапирање да трансформише скуп тренинг слика у вишедимензионални простор. Циљ је пронаћи оптималну хиперраван (границу одлуке) у вишедимензионалном простору која раздваја податке који припадају различитим класама. Уколико није могуће у потпуности поделити податке, треба наћи хиперраван која максимизује маргину, односно минимизује грешке у класификацији. На слици 9 је лево приказано више могућиг хиперравни, а десно оптимална хиперраван. Очекује се да хиперраван са већом маргином буде прецизнија приликом класификације непознатих података у односу на хиперраван са мањом маргином. Маргина представља најкраће растојање од хиперравни до најближег тренинг податка који припада било којој класи. 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. [17]



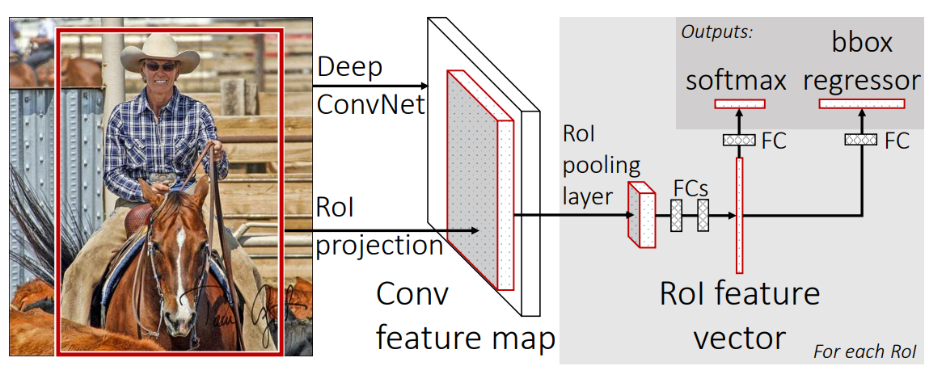
Слика 9: Лево: Могуће хиперравни. Десно: Оптимална хиперраван [17]

### R-CNN

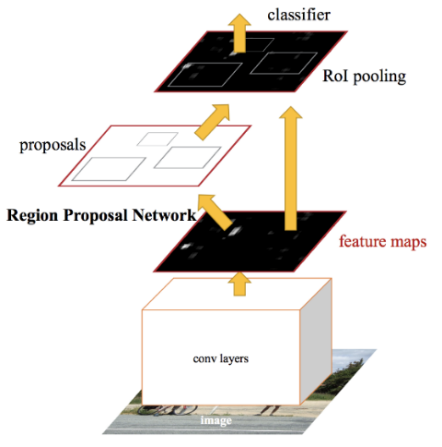
R-CNN (Region-based Convolutional Neural Networks) детектор из 2013. је један од првих који примењује дубоко учење за детекцију објеката. На слици 10 је приказано како модел функционише. Систем за детекцију објеката се састоји из три модула. Први модул применом селективног тражења издваја из улазне слике око 2000 предложених 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 класификује сваки од региона. За сваку класу се одређује score за сваки издвојени вектор фичера. Након класификације, примењује се пост-процесирање да се пречисте оквирни правоугаоници и елиминишу дупликати (bounding box regression). [18]



Слика 10: Начин функционисања R-CNN модела [18]

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. На слици 11 приказана је архитектура R-CNN-а. Мрежа која као улаз има комплетну слику и скуп предложених региона процесира слику и генерише конволуциону мапу фичера (conv feature map). 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 за сваки од региона. Сваки фичер вектор се затим користи за класификацију региона у једну од класа и врши се побољшање тачности ориналног обухватајућег правоугаоника коришћењем bounding box regressor-а. [19]

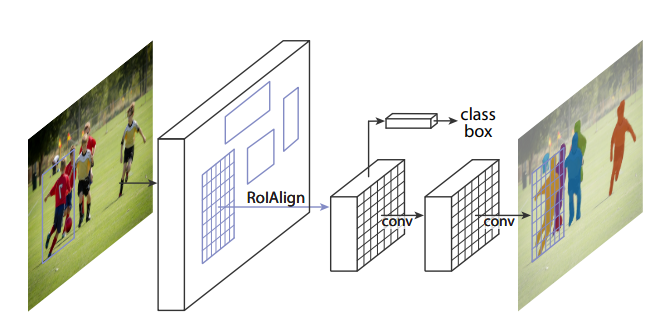
Слика 11: Архитектура Fast RCNN-а [19]

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 12 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. Систем је јединствена неуронска мрежа за детекцију објеката. [20]

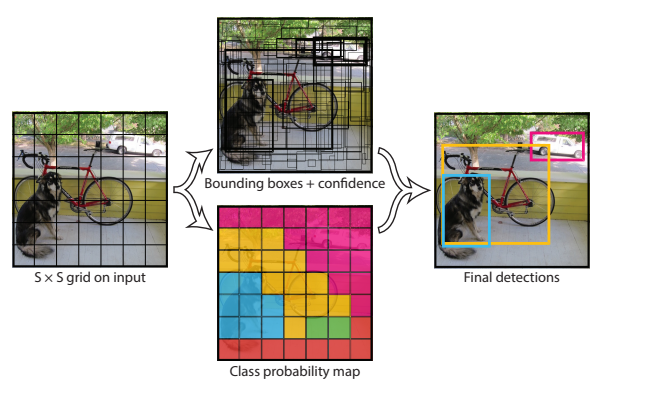
Слика 12: Faster CNN [20]

Mask R-CNN детектор из 2017. године је унапређење Faster R-CNN и приказан је на слици 13. 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). У овом моделу ROI Pooling модул је замењен поузданијим ROI Align модулом и додата је још једна грана из ROI Align модула. Ова додатна грана прихвата излаз ROI Align модула и прослеђује га у два конволуциона слоја, чији излаз је сама маска. [21]

Слика 13: The Mask R-CNN framework for instance segmentation [21]



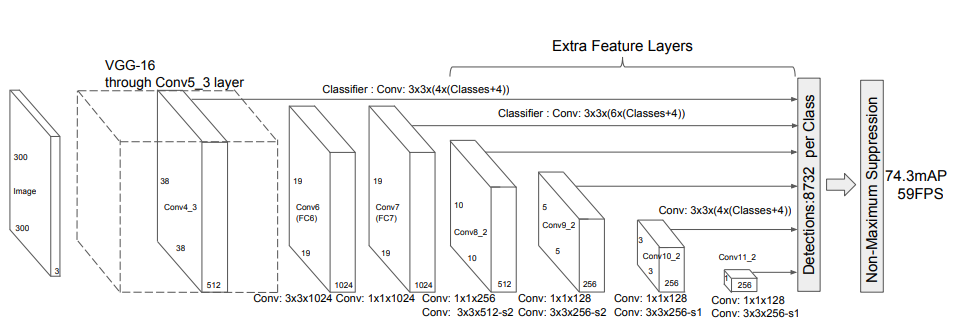
YOLO детектор

YOLO детектор је презентован у раду [22] који је објављен 2015. 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 овде 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“). 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 и поузданости Pr(Object) ∗ IOU за њих које представљају вероватноћу да правоугаоник садржи објекат. Ове поузданости казују колика је вероватноћа да правоугаоник садржи објекат. Сваки обухватајући правоугаоник има 5 предикција: x, y, w, h, и confidence (confidence score reflects how likely the box contains an object (objectness) and how accurate is the boundary box). 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) представља IOU[[4]](#footnote-4) (intersection over union) између предвиђеног обухватајућег правоугаоника и било ког ручно означеног правоугаоника у тест скупу слика који специфицира где се објекат налази. Свака ћелија грида такође предвиђа C условних вероватноћа за класе Pr (Classi | Object). 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: Pr(Classi|Object) ∗ Pr(Object) ∗ IOU = Pr(Classi) ∗ IOU. Ово нам даје вероватноћу да се та класа појављује у правоугаику, као и информацију о томе колико добро предвиђени правоугаоник представља објекат. Модел је имплементиран као конволуциона неуронска мрежа и евалуиран је на PASCAL VOC скупу података. Иницијални конволуциони слојеви издвајају фичере из слике, док потпуно повезани слојеви предвиђају излазне вероватноће и координате. Мрежа има 24 конволуционих слојева и 2 потпуно повезана слоја. Бржа варијанта мреже има мање конволуционих слојева (9) и мање филтера у тим слојевима. Како је при тренирању узето S = 7, B = 2 и C = 20 јер коришћени скуп података има 20 класа, излаз мреже је 7 × 7 × 30 тензор ((S, S, B×5 + C) = (7, 7, 2×5 + 20) = (7, 7, 30)). 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 од осталих детектора који раде у реалном времену. YOLO прави више грешака у лоцирању (поготово мањих објеката), али знатно мање лажних предикција када нема ничега на слици. Обучаван је на генерализованим репрезентацијама објеката, од природнних слика до уметничких. [22]

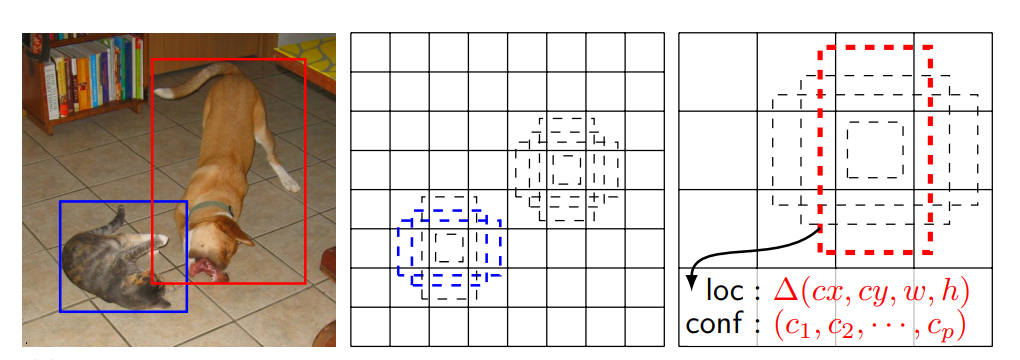
Слика 14: Модел YOLO детектора [22]

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 малих објеката у једној ћелији они неће бити детектовани.

SSD детектор

Овај метод [23] из 2016. године детектује објекте на сликама коришћењем дубоке конволуционе неуронске мреже и дизајниран је тако да ради у реалном времену. Достигао је преко 74% mAP и 59 FPS на PascalVOC скупу података. Детекција објеката обавља се у једном проласку кроз мрежу (Single Shot) и коришћена је MultiBox техника за bounding box регресију. Уводи неколико побољшања као што су мапе фичера у више величина и подразумевани правоугаоници (енг. default boxes). Конволуциона мрежа генерише неколико фиксних подразумеваних оквирних правоугаоника различитих величина и предикције (вероватноће и помераје правоугаоника) за присуство објеката у тим превоугаоницима коришћењем малих конволуционих филтера примењених на мапе фичера, што је праћено non-maximum suppression како би се добиле коначне предикције. Детекција објеката се састоји из два корака: издвајање мапа фичера и примена конволуционих филтера за детекцију објеката. Архитектура мреже приказана на слици 15 састоји се из два дела и заснована је на VGG-16 мрежи која се користи за класификацију слика. VGG-16 мрежа је одсечена пре класификационог слоја и њени потпуно повезани слојеви 6 и 7 су конвертовани у конволуционе слојеве. Затим следи скуп помоћних 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. SSD за издвајање мапа фичера користи VGG16, али могу се користити и други модели као основа.

Слика 15: Архитектура SSD детектора [23]

Посматрајмо слој Conv4\_3. Слика се дели на 38 × 38 ћелија, при чему се за сваку од њих генеришу 4 предикције за сваки од подразумеваних правоугаоника, тако да имамо укупно 38 × 38 × 4. Генерисање више предикција се зове multibox. Предикција се састоји од 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). Рачунање предикција се обавља применом 3 × 3 конволуционог филтера над сваком ћелијом. За класу уоквиреног објекта узима се класа са највећом вредношћу score-а. Овај поступак се обавља у сваком конволуционом слоју, при чему слојеви имају мапе фичера различитих димензија. Резолуција мапа фичера се смањује како се смањује величина слојева, што омогућава детекцију објеката различитих величина и значајно повољшава тачност детектора. Укупно има 6 конволуционих слојева, при чему се у неким од њих генеришу 4, а у неким 6 предикција. SSD укупно генерише 8732 (38\*38\*4 + 19\*19\*6 + 10\*10\*6 + 5\*5\*6 + 3\*3\*4 + 1\*1\*4 = 8732) предикција. Објекти могу бити различитих облика и потребно је да подразумевани правоугаоници покрију што више њих како би детектор могао да детектује што више типова објеката. Ови подразумевани правоугаоници су ручно одабрани и тежило се томе да их буде што мање како би цео процес био бржи. Као што је речено, за сваки подразумевани правоугаоник се генерише једна предикција. 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. На нивоу једног feature map слоја користе се исти поразумевани правоуганици за сваку ћелију и они су центрирани у тој ћелији, али различити слојеви користе различите подразумеване правоуганике услед разлике у резолуцији. SSD дефиниче фактор скалирања за сваки од слојева. Гледано са лева десно, фактор скалирања иде од 0.2 (или 0.1 некад) до 0.9. Комбиновањем ових фактора скалирања и жељених aspect ratios израчуванају се ширина и висина подразумеваних правоугаоника. Предикције се могу класификовати као позитивна и негативна поклапања, при чему се само позитивна користе за рачунање localization cost (the mismatch of the boundary box). Ако одговарајући подразумевани правоугаоник има IoU већи од 0.5 са ground truth, поклапање се сматра позитивним, а у супротном негативним. ground truth правоугаоници су они којима су ручно означени објекти на тренинг сликама. На пример, на слици имамо два подразумевана правоугаоника поклопљена са мачком и један са псом, који се сматрају позитивним преклапањима, а остали правоугаоници се сматрају негативним. На слици 16 је приказан и пример како SSD комбинује мапе фичера различитих величина и подразумеване оквирне правоугаонике да детектује објекте различитих величина и аspect ratios. 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). The localization loss је разлика у поклапању између ground truth правоугаоника и предиктованог оквирног правоугаоника, при чему се посматрају само позитивна поклапања. При рачунању се користи L2-Norm. The confidence loss је губитак услед предикције класе, при чему се посматрају и позитивна и негативна поклапања.g to the confidence score of the corresponding class (негативна поклапања се односе на позадинску класу). За рачунање се користи Categorical cross-entropy. 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, SSD, она се сортирају по calculated 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, а како би се обухватили објекти различитих величина и облика, свака тренинг слика може бити насумично узета као оригинал, као насумично парче или парче са IoU of 0.1, 0.3, 0.5, 0.7 или 0.9. The sampled patch will have an aspect ratio between 1/2 and 2. Then it is resized to a fixed size and we flip one-half of the training data. In addition, we can apply photo distortions. 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. 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. [24]

Слика 16 [23]

## Детекција и препознавање лица

Први корак било које обраде лица, попут препознавања лица, је детекција 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 људско лице, где је лоцирано и које је величине. [25] При проналажењу лица на слици неопходно је узети у обзир оријентацију лица, израз лица, осветљење, прекривеност лица (нпр. наочаре, коса, брада), резолуцију слике, комплексност позадине (присуство великог броја објеката) итд. Свако људско лице је јединствено и одређено је специфичним распоредом и величином његових саставних делова. Након детекције лица могу се издвојити карактеристичне тачке лица које означавају истакнуте деловe лица попут очију, обрва, носа, уста и образа. У рачунарском виду ове тачке се успешно примењују нпр. за поравнање лица, замену лица, препознавање лица, одређивање положаја главе, детекцију трептаја и праћење погледа. Постоје различити детектори истакнутих тачака лица, 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 (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. У овом поглављу биће дат преглед неколико најпознатијих детектора лица, као што су Haar cascades i DNN детектор из OpenCV библиотеке и HOG + Linear SVM и CNN из Dlib библиотеке i MTCNN (Multi-task Cascaded Convolutional Neural Network), а након тога ће бити дефинисана техника препознавања лица и дат преглед неколико најпознатијих метода за препознавање лица.

Haar cascades или Viola-Jones метод је напознатији детектор лица, веома је брз, нема велике хардверске захтеве и погодан за извршавање у реалном времену. Са друге стране, подлежан је лажно позитивним детекцијама и захтева ручно подешавање параметара. Треба га користити када је брзина извршавања приоритет и спремни смо да жртвујемо поузданост. Првенствено детектује фронтална лица и неће радити ако је лице прекривено нечим или није адекватно оријентисано. Систем користи само информације присутне на сивој слици (енг. grey scale image), без узимања у обзир додатних информација попут разлика у сликама у видео секвенци или боје пиксела на обојеним сликама, које могу повећати FPS. Са повећањем броја фичера расте поузданост, али и време извршавања. HOG + Linear SVM детектор ради тако што фичере издвојене преко хистограма оријентисаних градијената прослеђује SVM. Алгоритам се заснива на пирамидама слика и клизећем прозору. Поузданији је од Haar каскада и има стабилнију детекцију. Недостаци су што ради само са фронталним (и благо окренутим) лицима јер HOG дескриптор није инваријантан на промене у ротацији и углу гледања, 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. Dlib’s CNN (A Max-Margin (MMOD)) детектор је веома поуздан захваљујући самом дизајну алгоритма и квалитетном тренинг скупу. Модел је мали (мањи од 1MB), веома добро имплементиран и документован. Захтева да dlib буде инсталирана, конвертовање прозора за уоквиравање лица ако се користи OpenCV[[5]](#footnote-5), немогуће га је користити у реалном времену без GPU убрзања. Препорука је да се користи када не треба бринути о перформансама у реалном времену. Може детектовати лица која су под различитим оријентацијама, 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. 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 модел није превелики (око 10MB). Поузданији је од Haar cascades и HOG + Linear SVM, али мање поуздан од CNN MMOD из dlib библиотеке. Један од највећих недостатака је што је мање поуздан за детекцију лица са тамнијом бојом коже, али то се може решити тренирањем модела скупом слика који садржи слике људи различитих раса. Постоји и квантована Tensorflow верзија. 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. [26]

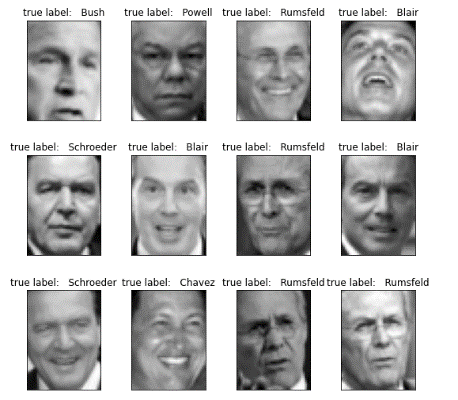
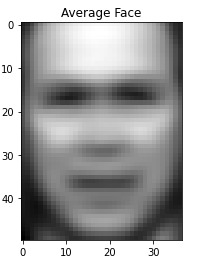
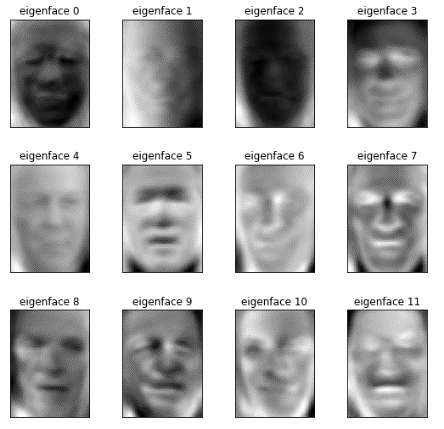
Препознавање лица је процес утврђивања коме одређено лице за слике припада (идентификације особе), односно, упоређивање неког лица са познатим лицима из базе података у циљу проналажења поклапања. [27] Не мора се у свим применама радити упоређивање са лицима из базе података јер је некада потребно само 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. Алгоритми се генерално разликују по понашању при екстремним условима као што су лоше осветљење, различити положаји лица, експресија лица, лица мале резолуције, прекривеност лица итд. У циљу побољшања резултата прпеознавања лица често се врши поравнање лица. Неке методе посматрају 3D модел лица, а једноставније методе се ослањају само на карактеристичне тачке (конкретно, на тачке које одређују очи) да би извршиле трансформације над сликом попут ротације, транслације и скалирања.

Код система за препознавања лица може бити велики проблем уколико неко злонамерно покуша да приступи систему представљајући се као неко други. Како се препознавање врши на основу података са камере, може се десити да неко подметне слику или видео друге особе. Уколико нема провере да ли је лице на камери реално, систем подметнута лица може препознати као валидна. Постоје различити приступи овом проблему, а неки од њих су: анализа текстура (рачунање Local Binary Patterns над регионом лица и коришћење SVM за класификацију реалних и лажних лица), анализа фреквенци (Фуријеов домен лица), анализа променљивих величина (попут вредности пиксела) 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. [28]

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 све више се користе алгоритми дубоког учења. Један од најзначајнијих алгоритама за развој препознавања лица је Eigenfaces, где је коришћена техника линеарне алгебре за редукцију димензија која се зове Principal Component Analysis (PCA). Затим је објављена 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. [29]

Eigenfaces

У раду [30] 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).

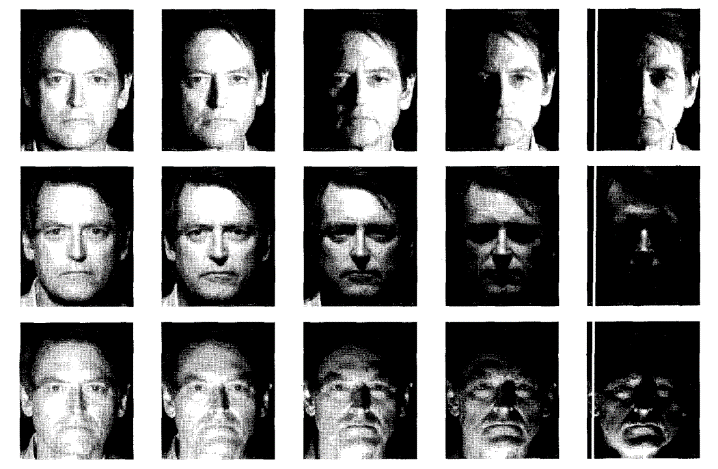
Алгоритам најпре обрађује скуп од М тренинг слика лица које су сиве, исте величине и поравнате тако да се кључни делови лица што више поклапају. Посматрамо слике димензија N×N (уопштем случају могу бити произвољне величине) које се могу линеаризовати и представити као вектор димензија N2, односно као 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 множењем матрице А и АT, 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“. 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: Лево: Тренинг лица. Средина: Средње лице. Десно: Eigenfaces [31]

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. [31]

Предност Eigenfaces алгоритма је што није рачунски захтеван и брзо се извршава Један од недостатака овог алгоритма је што захтева поравнање 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.

Fisherfaces

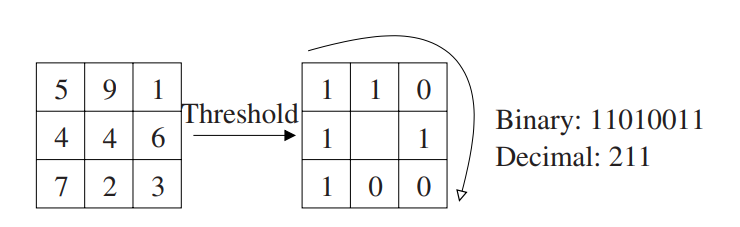
 Fisherfaces метод представљен 1997. године у раду [32] је уз Eigenfaces један од најпопуларнијих алгоритама препознавања лица. Овај алгоритам се такође заснива на редукцији димензија. Eigenfaces алгоритам користи анализу главних компоненти (PCA), док Fisherfaces користи линеарну дискриминантну анализу (LDA).

Слика 18: Слике лица при различитом осветљењу [32]

Сваки пиксел на слици посматран је као тачка у високодимензионом простору. 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 на којима су лица различитог идентитета. 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. Израчунају се матрице коваринсе између класа и унутар класа, а затим eigen вектори и еigen вредности. Бирају се eigen вектори са највећом eigen вредношћу. Као код Eigenfaces алгоритма, тренинг слике се пројектују у потпростор слика, улазна слика се такође пројектује у овај потпростор и упоређују се растојања. [32]

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 [33] 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 (слика 19). Затим се хистограм ових вредности користи као дескриптор текстуре.

Слика 19: LBP

Први корак у алгоритму 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). Вредности тежина су експериментално утврђене. Тежински хистограми се надовезују и формирају јединствени хистограм фичера који репрезентује слику лица. Препознавање лица се врши упоређивањем растојања. Улазно лице се обрађује као и тренинг лица (издвајају се LBP, додају им се тежине, конкатенирају се) и затим се примени k-NN (k=1) са растојањима како би се нашло најближе лице из тренинг података. [33]

Слика 20: Лево: Оригинална слика лица. Десно: тежинска шема за 7×7 ћелија [33]

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.

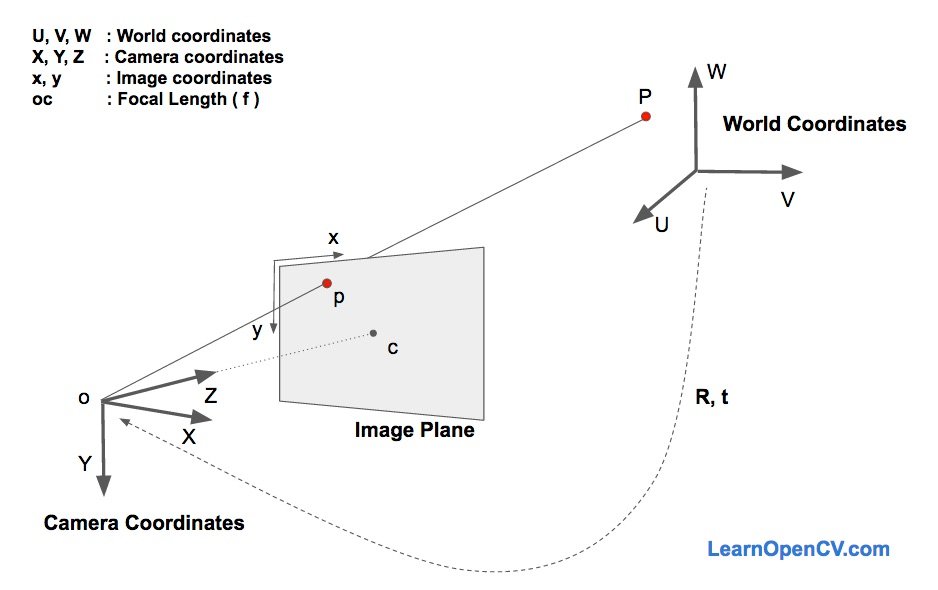
Дубоко учење

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 се обавља тако што се користе 3 различите слике (triplets), при чему две припадају истој особи, а трећа некој другој особи. Неуронска мрежа генерише 128-димензионе векторе за сваку од слика и 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. Мере генерисане за различите слике исте особе треба да буду скоро исте.

Слика 21: 28 мера лица [34]

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, што се може постићи коришћењем класификатора. [34]

## 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. 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, као и унутрашњи параметри камере (фокална дужина камере, оптички центар слике и параметри 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. Ми посматрамо 3 координатна система: 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 22: Координатни системи

На слици 22 о је центар камере, а раван која је приказана је раван слике. Нас интересују једначине које дају пројекцију p3D тачке P на раван слике.Нека је (U, V, W) позиција тачке P у светским координатама. Ако знамо ротациону матрицу R (3x3) и транслациони вектор *t* (3x1), 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 rij и (tx, ty, tz). 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 fx и fyфокалне дужине у x и y смеровима, а (cx, cy) 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:

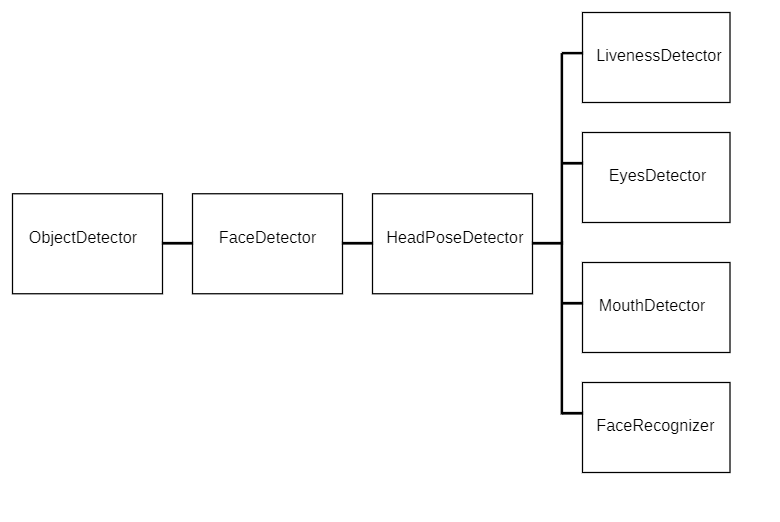
и може се решити коришћењем метода који се зове директна линеарна трансформација [(DLT)](https://en.wikipedia.org/wiki/Direct_linear_transformation). Овај метод се може користити кад год имамо проблем где је једначина скоро па линеарна, али је помножена непознатим фактором. Метод није баш поуздан зато што ротациона матрица R има три степена слободе, а матрица која се појављује у DLT решењу има 9 бројева. DLT решење не минимизује the correct objective function. У идеалном случају, ми 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. Када је процењена поза нетачна, можемо израчунати **грешку пројекције** између пројектованих 3D тачака и 2D карактеристичних тачака лица. 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 о могу се користити за одређивање позе. [35]

## Праћење покрета очију и усана

За праћење покрета очију и усана могу се искористити карактеристичне тачке лица.

# **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 комплетну платформу за полагање тестова, већ симулира полагање и надгледање теста укључивањем тајмера и прихватањем фрејмова са камере корисника. За рад апликације потребно је чувати податке о ученицима, а како се не би сви подаци чували у локалном фајл систему, коришћене су *Firebase*[[6]](#footnote-6) услуге. За конекцију са *Firebase*-ом коришћена је библиотека *firebase\_admin*, а за обраду слика (фрејмова) коришћене су библиотеке *OpenCV* и *Dlib*. Ове библиотеке су отвореног кода (енг. *open source*), веома су 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: омогућава додавање ученика у систем и позивање функције која покреће тест за одговарајућег ученика. Док је тест покренут обавештења се приказују у конзоли, а на снимку са камере који је приказан ученику исписује се колико је времена остало до краја теста. На слици 22 је приказан дијаграм који илуструје компоненте система које се користе за надгледање полагања. Издваја се неколико модула: *ObjectDetector* који контролише да ли се у кадру налази тачно једна особа и да ли нема недозвољених електронских уређаја попут мобилних телефона, *FaceDetector* који проверава да ли је детектовано једно 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 сликом ученика који је покренуо тест, *HeadPoseDetector* и *EyesDetector* који проверавају да ли су ученикова глава и поглед усмерени ка екрану и *MouthDetector* који проверава да ли ученик прича. Сваки од модула имплементиран је као класа, а из главног фајла који је улазна тачка апликације и који садржи функцију за симулацију теста, позивају се њихове функције за детекцију и валидацију. Оваква организација кода омогућава да измене у обради фрејмова буду локализоване у оквиру сваког модула.

Слика 22: Компоненте за надгледање полагања

На почетку и крају теста се комуницира са базом података преко посебног модула названог *Database*. За потребе реализације овог система чувани су само основни подаци о ученицима (идентификациони број, име, презиме и *email* адреса) и њихове слике, тако да је модел базе података максимално поједностављен. Слике се чувају у *Firebase Storage*, а линкови до слика заједно са осталим подацима о ученику у *Firestore Database*. Имплементиране су функције за додавање ученика у систем, учитавање информација о ученику и чување извештаја. *Firestore Database* је база података оријентисана ка документима (document-oriented), где су подаци организовани у документе, а документи у колекције докумената, тако да је направљена колекција *students* у којој је сваки ученик представљен једним документом. Сваки документ при креирању добија аутоматски генерисан јединствени *id*, који је искоришћен да се именује фолдер у *Firebase Storage* где ће бити чувани фајлови везани за тог ученика, као што су његова слика и видео извештаји. Приликом стартовања функције која симулира покретање теста учитава се слика ученика на основу његовог идентификационог броја и она се користи за касније упоређивање са сликом са камере у циљу препознавања лица. Претпоставка је да сваки ученик који је убачен у систем има слику на којој се види отприлике до мало испод рамена, лице му се може јасно уочити, гледа у камеру и не смеје се. Учитавају се и основни подаци о ученику како би му се приказала поздравна порука. Проверава се да ли учитана слика садржи лице, иницијализују се модули за детекцију говора и препознавање лица и омогућава се покретање теста.

Након покретања камере ученика, систем обрађује један по један фрејм. На слици 23 је приказан ток обраде фрејмова. Учитани фрејм прво пролази кроз детектор објеката. Уколико је детектована једна особа у кадру прелази се на детекцију лица и карактеристичних тачaкa лица у оквиру детектора лица. Уколико је и ова детекција успешна, издваја се 6 карактеристичних тачака потребних за одређивање положаја главе и у оквиру наредног детектора се проверава да ли је глава особе усмерена ка екрану. Уколико то јесте случај, следи издвајање карактеристичних тачака очију и поравнање лица, након чега се посматрају карактеристичне тачке поравнатог лица. Тачке којима су одређене очи прослеђују се детектору који проверава да ли је лице реално бројањем трептаја и детектору који праћењем зенице ока проверава да ли особа гледа право у екран. Тачке којима су одређене усне прослеђују се детектору говора који прати да ли су уста отворена или не. Затим се поравната слика и карактеристичне тачке прослеђују модулу за препознавање лица.

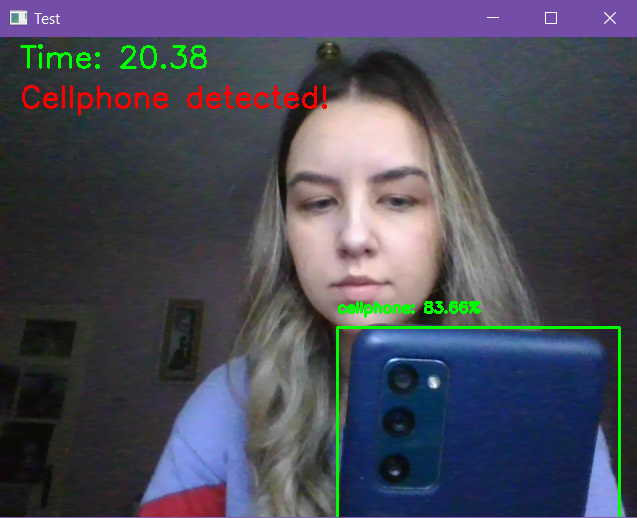
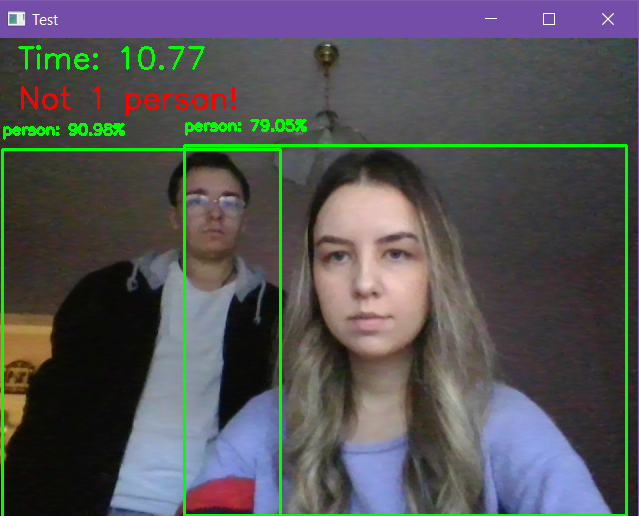
Обрада фрејмова се прекида када прође онолико времена на колико је подешено трајање теста и тада се у видео фајл бележи извештај који се касније може прегледати ради потврђивања детектованих нерегуларности. Генеришу се два фајла: један који садржи комплетан снимак полагања са означеним нерегуларним ситуацијама, и други, који садржи само делове полагања где је било нерегуларности. Сваки од појединачних детектора има свој бафер у коме чува прозоре фрејмова (прозором се сматра одређени број сукцесивних фрејмова). Детектор означава фрејмове прозора који се сматрају нерегуларним. Да би се у посебном баферу главног програма издвојили само нерегуларни фрејмови, детектор „пресипа“ своје нерегуларне прозоре у бафер у главном програму намењен том детектору. На крају теста, ови бафери из главног програма се спајају у јединствени бафер који садржи све нерегуларне ситуације. С обзиром на то да може да се деси да су се нерегуларни прозори из различитих детектора преклапали, односно да се исти фрејмови налазе у прозорима различитих детектора, из овог бафера су елиминисани дупликати. Бафер се сортира тако да фрејмови буду временски уређени и његов садржај се уписује у видео фајл који садржи само нерегуларне ситуације. Направљена је посебна класа која представља један фрејм и која има три атрибута: слику, редни број фрејма и поруку о нерегуларностима која ће бити исписана на фрејму. Сваки фрејм има и испис вредности тајмера у тренутку обраде тог фрејма како би се знало када се тачно десила нека нерегуларна ситуација. Атрибут који чува информацију о редном броју се користи приликом сортирања елемената коначног бафера. Пошто у једном фрејму може истовремено бити више нерегуларности, сваки од детектора ажурира овај атрибут поруке дописивањем специфичног текста. У наставку ће бити детаљније описан сваки од коришћених модула за детекцију.

## Детекција објеката на камери

При одабиру модела за детекцију објеката треба имати у виду конкретан случај коришћења и потребе саме апликације. Један од најбољих и најпопуларнијих 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 који је коришћен у оквиру модула за дубоко учење из OpenCV библиотеке. Овај модул нуди и могућност извршавања на GPU, али је оптимизован за извршење на CPU. Омогућава коришћење истренираних мрежа из framеwork-a за дубоко учење попут TensorFlow, Torch и Caffe. У пројекту је коришћена TensorFlow имплементација [36] модела који комбинује MobileNet архитектуру и Single Shot Detector (SSD). Модел је трениран на COCO [37] скупу података који садржи 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.

У оквиру детектора објеката (модул назван ObjectDetector) биле су од значаја следеће класе објеката: *person*, *laptop* и *cellphone* зато што је циљ био детектовати да ли постоји тачно једна особа у кадру и да ли се користе недозвољени електронски уређаји попут мобилних телефона и лаптопова. Учитан је текстуални фајл који садржи лабеле за сваку класу објеката из COCO скупа слика и парсиран је тако да се од њега направи низ који садржи називе класа које се могу детектовати. Мрежа је учитана позивом функције cv2.dnn.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 функција cv2.dnn.blobFromImage промењена на 300x300 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 ћемо множењем тих бројева са оригиналном ширином и висином слике добити 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.

Главна функција у којој се учитавају фрејмови са камере прослеђује овом детектору један по један фрејм за обраду. С обзиром на то да нерегуларна ситуација (у кадру нема ниједне особе или има више од једне особе, детектован је мобилни телефон или лаптоп) неће трајати само један фрејм, већ више узастопних фрејмова, а и како би се избегле случајне лажне детекције, посматрају се „прозори“ фрејмова. За сваку од 3 класе од интереса направљен је по један бафер прозора, као и помоћни бафери и променљиве. Бафер прозора чува све фрејмове који припадају једном прозору. 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 и „пресипа“ у главни бафер који садржи нерегуларне ситуације везане за ту класу објеката на нивоу комплетног теста и који се налази у главној класи апликације. На крају сваког прозора прозор бафер се ресетује. Постоји још један бафер који чува узастопне нерегуларне фрејмове у једном прозору. Овај бафер 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 15 елемената и ако има ти фрејмови се пребацују у главни бафер. Након тога се прелази на следећи прозор. Ово је интерна логика детектора објеката и промене су видљиве кроз ажурирање бафера из главне класе који су додељени овом детектору (за сваки тип објеката по један). Резултат обраде сваког појединачног фрејма може утицати на преостале детекторе у систему који обрађују податке са лица особе зато што ако нема само једне особе у фрејму, нема потребе прослеђивати тај фрејм даље - он ће представљати прекид прозора других детектора и главна класа ће позвати одговарајуће функције за ресетовање осталих детектора. Обезбеђена је и функција која ће око детектованог објекта исцртати оквирни правоугаоник у нерегуларним фрејмовима како би се јасно обележио (слика 23). Ако особа прислони телефон уз главу док разговара велика је вероватноћа да телефон неће бити детектован, али ако га држи као на слици 23 (лево) и покушава да нешто прочита, биће детектован.

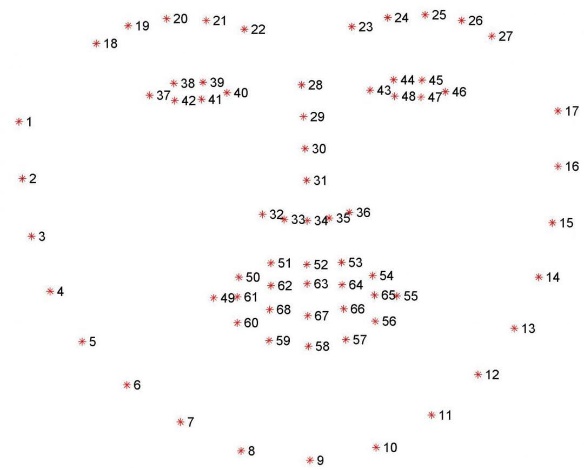
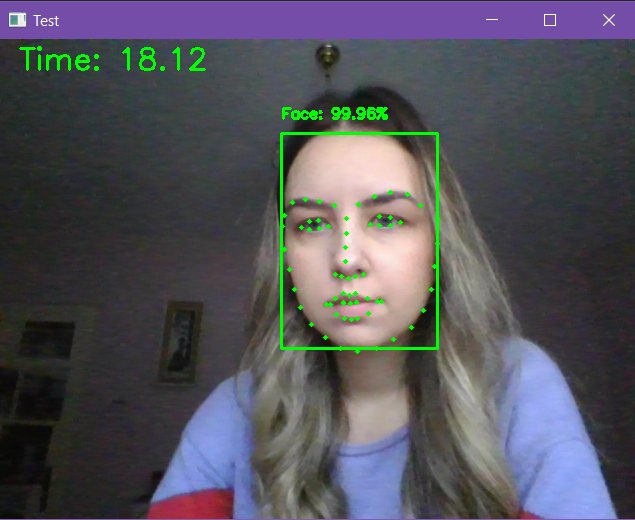


Слика 23: Лево: Детекција мобилног телефона. Десно: Детекција 2 особе

## Детекција лица и карактеристичних тачака лица

За детекцију лица у оквиру класе FaceDetector коришћен је такође OpenCV DNN модул. Конкретно, учитана је квантована TensorFlow верзија [37] детектора лица. Моделу треба проследити слике димензија 300x300 над којима је примењено одузимање средње вредности тренинг скупа слика. Поступак обраде детекција је врло сличан као код детекције објеката. Нерегуларним фрејмовима се сматрају фрејмови на којима се не налази тачно једно лице. Овај детектор има еквивалентне бафере 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 узастопних невалидних фрејмова и ако јесте пребацује их у бафер намењен овом детектору из главне класе. Ако то није случај, проверава да ли је обрађено више од две трећине прекинутог прозора. Ако јесте, проверава да ли је више од половине тих фрејмова нерегуларно и ако то јесте случај пребацује се прекинути прозор у главни бафер. Слично као код детектора објеката, када овај детектор утврди да на слици не постоји тачно једно лице, ресетоваће се детектори који су хијерархијски испод и обрађују лице.

Детекција карактеристичних тачака лица на лицу обављена је унутар класе LandmarksDetector коришћењем модела који детектује 68 тачака из Dlib библиотеке [38]. Детектују се координате тачака које се мапирају на структуру лица и оне су приказане на слици 24.



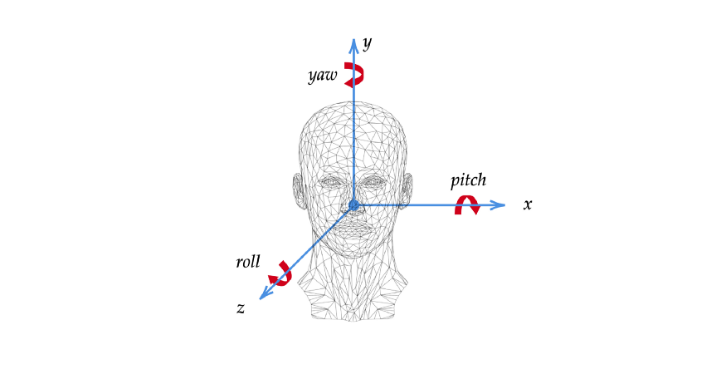
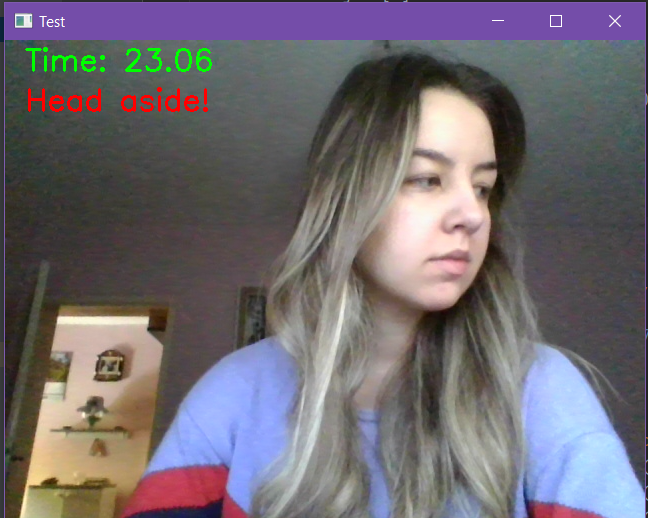
Слика 25: Детектовано лице и карактеристичне тачке

Слика 24: 68 карактеритичних тачака [39]

Детектор као повратну вредност има *shape* објекат који садржи (x, y) координате 68 карактеристичних тачака лица. У оквиру класе овог детектора имплементиране су функције које враћају специфичне тачке лица које се користе у наставку обраде 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).

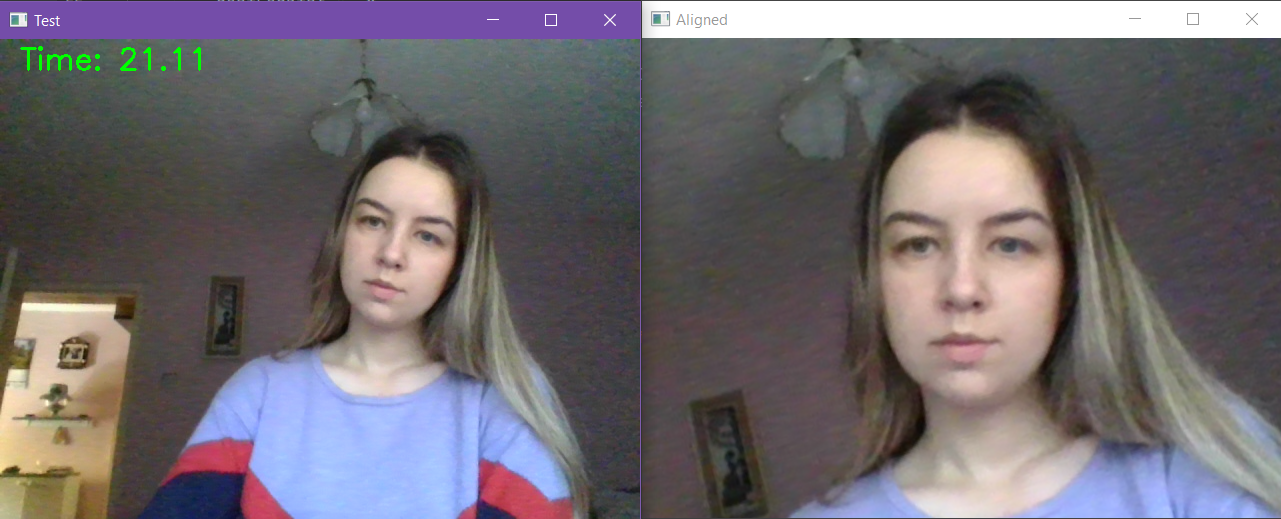
## Детекција окретања главе

Праћење положаја главе ученика је имплементирано коришћењем карактеристичних тачака лица и неколико функција из OpenCV библиотеке. Потребно нам је да знамо да ли је ученикова глава окренута сувише лево, десно, горе или доле и ако јесте да то забележимо у одговарајући бафер. 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, а карактеристичне тачке које се користе су добијене из детектора карактеристичних тачака 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 је solvePnP и она имплементира неколико алгоритама који могу бити селектовани приликом њеног позива, а подразумевани је SOLVEPNP\_ITERATIVE који је DLT решење праћено Levenberg-Marquardt оптимизацијом. 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. **Функција враћа ротациони и транслациони вектор који трансформишу** 3D тачке објекта у координате камере. [35] 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. [40] На слици је приказана нерегуларна ситуација где је угао ротације око у-осе већи од граничног.

## Поравнање лица

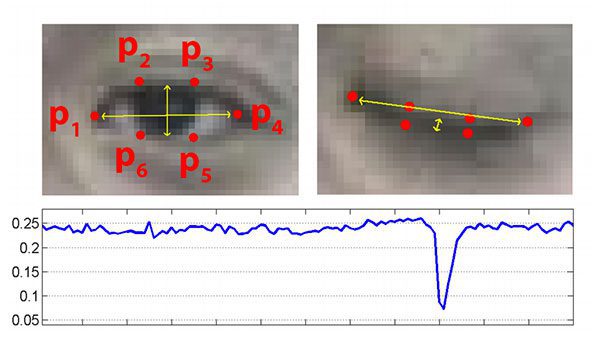
Поравнање лица се ради како би се побољшали алгоритми који ће се примењивати над лицем. У нашем случају то су препознавање лица, праћење погледа и детекција говора. Желимо да лице буде центрирано на слици, заротирано тако да очи буду на истој хоризонталној линији и да буде скалирано тако да величина свих лица буде приближно идентична. Слика са поравнатим лицем се враћа позивајућем програму и даља обрада се врши над овом сликом.

Коришћена је афина трансформација. Афине трансформације се користе за ротацију, скалирање, трансалцију итд. и можемо све три трансформације објединити у један позив cv2.warpAffine функције којој се као један од параметара прослеђује ротациона матрица, М. На почетку се задаје жељена (x, y) позиција левог ока. Координате су углавном изражене у процентима из опсега 20-40%. (у овом пројекту узето је 0.43-isprobati sa 0.4 i primeniti nad ulaznom slikom). 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). Сада можемо израчунати жељену дистанцу између два ока по хоризонтали и помножити је са ширином слике јер је добијена вредност из опсега 0-1. Дељењем жељене дистанце са тренутном дистанцом по хоризонтали можемо израчунати фактор скалирања. Потребно је наћи централну тачку која се налази између два ока јер ће се око ње вршити ротација. За рачунање ротационе матрице 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 ширине слике, и друга у којој се налази висина слике помножена жељеном y координатом левог ока. Транслациона компонента матрице је ажурирана тако што су ове две вредности одузете од одговарајућих координата 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, у нашем случају INTER\_CUBIC. [41]

Провера да ли је лице реално

У овом пројекту коришћена је детекција трептања ученика за проверу да ли је ливе „живо“, односно да ли није подметнута слика лица. Детектовани су и бројани трептаји ока у одређеном временском интервалу.

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 бела површина у оку и одређивање да ли бела површина нестаје на одређени временски период, eye aspect ratio 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:

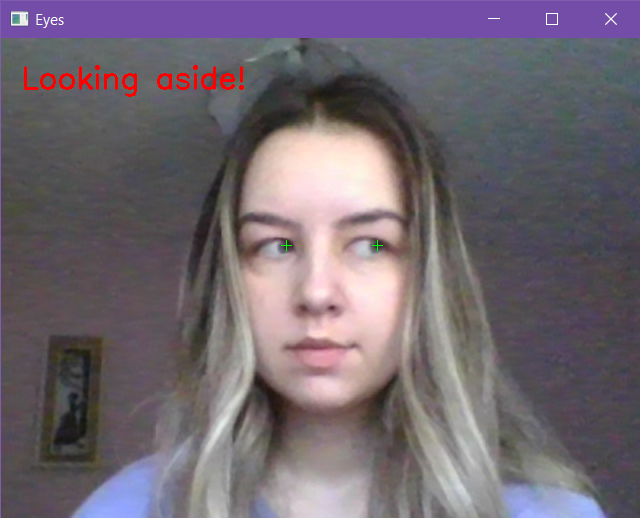
где су p1, …, p6 значајне тачке које одређују око. Ова величина је приближно константна док је око отворено, а нагло опада до нуле када се деси трептај. Коришћењем ове метода избегавамо технике процесирања слике и ослањамо се на растојања између одговарајућих тачака. На графику се може видети EAR кроз време на видео снимку. Вредност је константна, затим нагло опада до нуле, а онда поново расте, што индицира да се десио трептај. [42]

Слика: Детекција трептаја [42]

Трептај ока ће трајати неколико фрејмова. Три узастопна фрејма која имају EAR мањи од граничне вредности (0.25) 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, узет је прозор од 30 секунди јер се претпоставља да ниједан тест неће трајати толико кратко. Очекује се да ће у том периоду особа трепнути између 6 и 11 пута тако да су прозори који имају бројач трептаја ван овог опсега сматрани нерегуларним. На крају сваког нерегуларног прозора ти фрејмови се пребацују у главни бафер намењен овом детектору. Услед неправилности у неком од детектора изнад, прозор и бројачи се ресетују.

## Праћење погледа

Ученик би током полагања теста требало да гледа право у екран и да не скреће поглед лево и десно. У модулу за праћење погледа детектују се периоди када постоји дуже гледање у једну или другу страну, а 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 улазна 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 threshold-а како би се истакла 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 cv2.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 threshold 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 cv2.bilateralFilter над издвојеним оком како би се уклонио нежељени шум, затим се направи кернел 3х3 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 threshold-а користе се функције cv2.findContours и cv2.moments како бисе нашле координате центра зенице. [43]

## Детекција говора

Причање је реализовано 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. Један бафер представља прозор фрејмова и његова величина је лимитирана на 30 елемената. У оквиру прозора 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 главни бафер 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 фрејмове у којима су уста узастопно 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 и пребацује у главни бафер.

## Препознавање лица

У сврху препознавања лица коришћена је неуронска мрежа. Постоји више мрежа које су већ истрениране и доступне за коришћење, тако да није било неопходно радити тренирање, већ је искоришћен постојећи модел, који је коришћен и у face\_recognition [44] библиотеци. Овај модел има accuracy од 99.38% на *Labeled Faces in the Wild* скупу слика.

За препознавање лица као прозор коришћен је бафер величине 50 фрејмова 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 постављена ученикова слика које је учитана из базе и израчунати су encodings позивом функције net.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, лице није препознато, тај прозор се сматра невалидним и пребацује се у бафер намењен овом детектору. Нема потребе упоређивати улазну слику са свим сликама из базе. При ресетовању овог детектора, уколико се до момента ресета детектовао невалидни фрејм, прозор се пребацује у одговарајући бафер и пријављује се нерегуларност.

# **Закључак**

Софтвер за полагање тестова је све напреднији и све више коришћен како у образовним установама, тако и за потребе тестирања у различитим компанијама и организацијама. Вештачка интелигенција и машинско учење у великој мери доприносе реализацији тестирања на даљину. Софтвер за аутоматско надгледање полагања онлајн тестова у доброј мери омогућава детекцију неправилности које можда не би биле детектоване при стандардном тестирању у учионици. Појединци се могу надгледати детаљније и надгледање је константно за разлику од ситуације у учионици када наставник у неком тренутку скрене поглед и створи прилику ученику за преписивање, међутим, ученици/студенти ће вероватно увек моћи да пронађу неки начин да варају. Предности оваквог начина тестирања су и 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. Са друге стране, овакав софтвер може бити скуп.

Онлајн преписивање може бити спречено на пример увођењем временског ограничења за одговор на питање како ученик не би имао времена да потражи помоћ. Може се и онемогућити враћање на претходно одговорено питање. Како ученици не би делили међусобно питања и решења након што су завршили тест, може се захтевати да сви полажу тест истовремено. Међутим, ове мере некада нису довољне и студенти ипак пронађу начин да преписују. Једна од најчешћих ситуација је да неко други ради тест уместо особе која треба да полаже, што се може превазићи методама за препознавање лица тако што ће се повремено бележити слика са камере и проверавати.

Главне предности софтвера за полагање онлајн тестова су доступност, мање ангажованог особља, константно надгледање током израде и могућност напредније анализе резултата.

Увек је било преписивања и могу се пронаћи начини да се превари дежурни наставник или софтвер који се користи за надгледање онлајн полагања. Међутим, ако узмемо у обзир напредак технологије и развој машинског учења, све више покушаја преваре може се детектовати и санкционисати. И даље се не можемо у потпуности ослонити на аутоматско надгледање и потребно је укључити људски фактор, али чињеница је да су системи све напреднији и да имају све већу примену како у образовању, тако и у осталим гранама друштва. У овом раду дат је преглед техника које се могу приметити у обради података са камере приликом тестирања и реализован је прототип система који примељује неке од њих.

# **Литература**

1. <https://www.testreach.com/blog-post/proctoring-online-exams.html?fbclid=IwAR0r8C4Cg4BLoVv-VJdbadDFxZk8qAiA_HHEkq7mLrEGkpa-U_7iah8jG10>
2. <https://www.techjockey.com/blog/proctoring-software-for-online-exam>
3. https://www.masterteachingonline.com/can-an-online-exam-detect-cheating/?fbclid=IwAR0CTcD3-2ixdQQaBXkrii\_eVHSyffVz2AZ4P08q0tc4cSRFCm0-VVLCOPs
4. <https://honorlock.com/>
5. <https://proctoredu.com/industries/corporate/pre-employment>
6. <https://pages.mettl.com/faq-online-examination#Cheating-Prevention>
7. <https://www.speedexam.net/>
8. <https://examus.com/>
9. https://en.wikipedia.org/wiki/Computer\_vision
10. <https://en.wikipedia.org/wiki/Object_detection>
11. https://viso.ai/edge-ai/ai-hardware-accelerators-overview/
12. <https://viso.ai/deep-learning/object-detection/>
13. https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf
14. <https://en.wikipedia.org/wiki/Haar_wavelet#Haar_functions_and_Haar_system>
15. <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>
16. <https://learnopencv.com/histogram-of-oriented-gradients/>
17. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
18. <https://arxiv.org/pdf/1311.2524.pdf>
19. <https://arxiv.org/abs/1504.08083?so>
20. <https://arxiv.org/abs/1506.01497>
21. <https://arxiv.org/abs/1703.06870>
22. <https://arxiv.org/abs/1506.02640>
23. <https://arxiv.org/abs/1512.02325>
24. <https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06>
25. <https://en.wikipedia.org/wiki/Face_detection>
26. https://pyimagesearch.com/2021/04/26/face-detection-tips-suggestions-and-best-practices/
27. <https://en.wikipedia.org/wiki/Facial_recognition_system>
28. https://pyimagesearch.com/2019/03/11/liveness-detection-with-opencv/
29. <https://pyimagesearch.com/2021/05/01/what-is-face-recognition/>
30. https://sites.cs.ucsb.edu/~mturk/Papers/mturk-CVPR91.pdf
31. https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/
32. <https://web.ece.ucsb.edu/~hespanha/published/faces.pdf>
33. https://link.springer.com/content/pdf/10.1007/978-3-540-24670-1\_36.pdf
34. <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>
35. <https://learnopencv.com/head-pose-estimation-using-opencv-and-dlib/#disqus_thread>
36. <https://github.com/opencv/opencv/wiki/TensorFlow-Object-Detection-API>
37. <https://github.com/opencv/opencv/tree/4.x/samples/dnn/face_detector>
38. <http://dlib.net/face_landmark_detection.py.html>
39. <https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/>
40. <https://medium.com/analytics-vidhya/real-time-head-pose-estimation-with-opencv-and-dlib-e8dc10d62078>
41. <https://pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/>
42. <http://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf>
43. <https://github.com/antoinelame/GazeTracking>
44. https://github.com/ageitgey/face\_recognition#face-recognition

1. LMS (Learning Management System) платформа је софтвер који се користи за креирање, администрацију и праћење тренинга и обука. [↑](#footnote-ref-1)
2. AI акцелератори су врста специјализованих хардверских акцелератора који се користе да убрзају аплиакције које користе вештачку интелигенцију и машинско учење. Неки од примера су VPU (енг. Vision Processing Unit), GPU (енг. Graphics Processing Unit) и TPU (енг. Tensor Processing Unit). [11] [↑](#footnote-ref-2)
3. Haar-ов талас је секвенца функција квадратног облика које формирају фамилију таласа или базу. [14] [↑](#footnote-ref-3)
4. IoU (пресек над унијом) је метрика за евалуацију тачности детектора објеката. Представња количник површине преклапања предиктованог правоугаоника и ручно означеног правоугаоника из тест скупа података и површине уније та два правоугаоника. Добром предикцијом се сматра IoU већи од 0,5. [↑](#footnote-ref-4)
5. У OpenCV-у, оквирни правоугаоник је представњен уређеном четворком (x, y, w, h), док је у dlib-у представљен објектом правоугаоника са left, top, right, and bottom атрибутима. [↑](#footnote-ref-5)
6. Firebase је Гуглова (енг. Google) платформа за развој апликација која нуди различите сервисе попут база података, аутентификације и аналитике. [↑](#footnote-ref-6)