

Óbudai Egyetem
Neumann János Informatikai Kar
Szoftvertchnológia Intézet



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

WATCHR – AZONOSÍTÁS ARCFELISMERÉSSEL

Szerző: **Varga Miklós**
mérnök informatikus szak, IV. évf.

Konzulensek: **Dr. Vámosy Zoltán**
egyetemi docens
Dr. Sergyán Szabolcs
egyetemi docens

Tartalomjegyzék

Bevezető.....	4
1. Irodalmi összefoglaló.....	5
1.1 Arcdetektáló technikák.....	5
Tudás alapú technikák.....	5
Jellemző alapú technikák	5
Mintaillesztés alapú technikák.....	6
Megjelenés alapú technikák	6
1.2 Arcfelismerő technikák	8
Geometriai tulajdonságokon alapuló módszerek.....	8
Sajátarc módszer.....	8
Sablon alapú illesztés	8
Neurális hálózat alapú modell	8
Gabor szűrők.....	9
Háromdimenziós technikák	9
Edge mapping.....	9
Line Edge Mapping.....	10
2. Hasonló rendszerek.....	11
Cerberus.....	11
Myra	11
Face Detection	12
Bananascreen.....	13
Értékelés.....	14
3. Saját rendszer bemutatása.....	15
3.1 Célpontosítás.....	15
3.2 Rendszervázlat	15
3.3 Modulok részletes kifejtése.....	16
Előfeldolgozás	16
Az arcdetektálás	17
Arcfelismerés	18
4. Tesztelések, eredmények bemutatása.....	23
Az előfeldolgozás eredményessége.....	23
A detektálás eredményei.....	23
A felismerés eredményei.....	25
5. Összegzés, jövőbeli tervek.....	27
Ábrajegyzék	28

Irodalomjegyzék	29
-----------------------	----

Absztrakt

A célom egy olyan program létrehozása, amely képes lehet leváltani a hagyományos jelszó alapú belépő rendszereket. Az azonosításhoz az arc biometria tulajdonságait használom fel, ami csaknem annyira egyedi jellemző, mint az ujjlenyomat.

Az arcfelismerés problémájával, már évek óta foglalkoznak, de még mindig nincs teljesen megoldva, hisz mindenképp számítani kell bizonyos mértékű mérési hibákra. Azonban a gépek teljesítményének növekedésével, egyre pontosabb és teljesítmény-igényesebb algoritmusok válnak futási időben használhatóvá. Továbbá a kamerával felszerelt mobil eszközök megnövekedett száma miatt egyre nagyobb kereslet is van az ilyen programok iránt.

Többféle lehetőség is van komplex alakok megkeresésére a képen, illetve azokból való információ kinyerésére. A programom először egyszerűsíti a kapott képet, majd pedig különböző szűrők segítségével a „nem arc objektumok” arc tulajdonságait elnyomja annyira, hogy a későbbi lépésekben a detektáló modul ne detektálja őket. Mind a detektáláshoz, mind a felismeréshez holisztikus algoritmusokat használok. Detektálásnál a Viola-Jones féle objektumdetektáló algoritmust használom, amely kaszkádláncba rendezett maszkokkal vizsgálja a kép Haar jellemzőit. A detektált arcok felismeréséhez két algoritmust használok, ezeknek a kombinált eredményeiből születik a végső döntés, így ezek egymás kontroljaként tudnak működni. Az egyik használt algoritmus a statisztikában használt főkomponens analízisen alapuló saját arcok módszere, a másik pedig az éleket vizsgáló edge map algoritmus.

Bevezető

Manapság az informatikában az egyik legfontosabb tényező a biztonság, és az egyre biztosabb azonosítás. Egy személy, felhasználó azonosítására több lehetőség van, például a tudásalapú rendszerek (jelszó), birtok alapú rendszerek (mágnes kártya), és az egyedi tulajdonságok alapú (biometria). A problémát az okozza, hogy a jelszavakat kezelni nehéz: hosszú, értelmetlen karaktersorozat minden rendszerhez különböző, megjegyezni nehéz, ha leírjuk, azt ellophatják. A tárgyalapú azonosítókkal is az a probléma hogy relatív könnyű őket ellopni, így legbiztonságosabbnak a biometrikus azonosítás tűnik. Először is érdemes tisztázni, mi is a biometria. „A biometria: egy személyt megkülönböztetni a többtől élettani vagy magatartási tulajdonságai alapján. A megkülönböztető jellemzőket nevezzük biometriai jellemzőknek. Egyes szakírók a biometriai jellemzők körébe az automatikusan felhasználható biometriai jellemzőket értik, így nem sorolják ezek közé pl. a DNS-t, ami a köztudatban is a legerősebb ilyen jellemző” [Reco 12]. Biometrikus azonosításra akkor alkalmas egy jellemző, ha általános (mindenkire alkalmazható), állandó (időben nem, vagy csak lassan változik) és technikailag rögzíthető [Reco 12]. Jelenleg egyetlen technika sem elégíti ki az optimális rendszer követelményeit (gyors, robosztus, pontos, megbízható), ezért céltól és felhasználástól függően kell megválasztani a használandó rendszert, vagy vegyíteni azokat.

A projekt célja egy olyan azonosítható tulajdonság használata, melynek vizsgálata nem okoz kellemetlenséget, és a vizsgálat maga nem feltűnő. Ezen kritériumoknak az arcvizsgálat megfelel. Normál kamerás arcdetektálásnak alapvetően kétféle megközelítése van. Az egyik megközelítés kimondottan videó feldolgozással foglalkozik, itt a környezet változása, az emberek mozgása, a könnyen felismerhető mozgásminták, emberi cselekvések szolgálnak segítségül. A másik megközelítés, amit én is használni szándékozok, alapvetően fényképek feldolgozásával foglalkozik. Fontos szempont még, hogy a normál kamerás arcfelismerést egy egyszerű webkamerával szándékozom megoldani, így tehát az olyan megoldások, melyeknek nagy felbontású, jó minőségű képekre van szükségük, a háttérbe szorulnak.

1. Irodalmi összefoglaló

1.1 Arcdetektáló technikák

Az arcdetektáló módszereket, működésük alapján négy csoportba lehet besorolni [Yang02]: tudás alapú-, jellemző alapú-, mintaillesztés alapú- és megjelenés alapú technikák.

Tudás alapú technikák

Ezen módszerek az általános emberi tudáson alapszanak, hogy milyen is az arc. Ilyen tudás például, hogy egy arcról készült képen, a szemek körül lesznek a legkisebb intenzitások, míg az orrnál a legnagyobb. Ilyen ismeret alapú technika a top-down modell is. „A top-down módszer abból indul ki, hogy különböző felbontáson különböző arcmentákat kapunk ugyan arról a képről. Tehát az algoritmus először az eredeti kép egy kis felbontású változatán arcmentákat, majd növelve a felbontást az adott szinthez tartozó arcmentákat keresi és összehasonlítja az előző szint eredményeivel, egészen addig, amíg el nem éri az eredeti képet. Ez a módszer főleg olyankor működik, amikor egyetlen arc van a képen” [Kövé05]. A módszer szemléltetése az 1. ábrán látható.

Jellemző alapú technikák

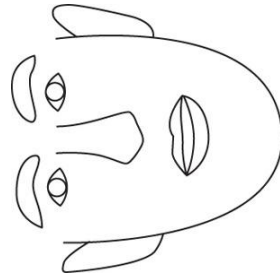
Az ilyen módszerek lényege, hogy először megkeresik az arc jellemző részeit (szem, száj, orr), majd ezekből csoportokat alkotva detektálják az arcot. „Ez történhet különböző képszűrők és szín/kontrasztjellemzők fölhasználásával, vagy pedig mintaillesztéssel” [Kövé05].



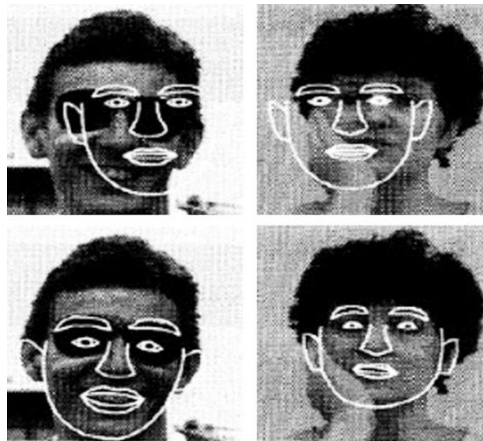
1. ábra Egy példa a top-down modell működéséről



2. ábra Tudásalapú technikáknál használt tipikus kép



3. ábra A keresett sablon



4. ábra Vizsgált képek

Mintaillesztés alapú technikák

Ezen technikák azon alapulnak, hogy egy előre elkészített arcmintát (lásd 3. ábra) keresünk a képen (4. ábra). Ez lehet egy éllel megadott sablon. Az ilyen módszerek hátránya, hogy annak a sablonnak, amit használni akarunk, nagyon reprezentatívnak kell lennie, mivel különböző népcsoportoknak, különböző közös jellemzőik vannak. Továbbá az előfeldolgozásnak nagyon jónak kell lennie, mivel a fényviszonyokra az ilyen módszerek érzékenyek. Mindemellett ezek az algoritmusok lassúak, nem képesek valós időben működni, mivel különböző orientációkat és skálázásokat kell végig próbálni az egész képen.

Megjelenés alapú technikák

Egy előre elkészített tanító mintákat tartalmazó halmaz alapján készítenek egy „modellt”, és ennek a modellnek megfelelő részeket keresnek a képen. Az ilyen módszereknél jellemzően neurális hálózatot is használnak.

Rowley, Baluja, és Kanade 1998-ban előálltak egy neurális hálózat alapú arcdetektáló rendszerrel. A módszer lényege, hogy a vizsgált képet kisebb részekre bontja, és azokat vizsgálva eldönti egy neurális hálózat, hogy van az eredeti képen arc vagy sem [Rowley98].

Viola és Jones 2001-ben egy hasonló rendszert alkotott azzal a különbséggel, hogy ők az Adaboost eljárás segítségével kiválasztották és kaszkádláncba fűzték azokat a Haar jellemzőket, melyek a legjobban jellemzik az arcot. Módszerük nagyon hatékony, rendkívül jó találati aránnyal rendelkezik és gyors [Viola01].

Az ilyen módszerek jól működnek szemből, vagy profilból, betanítástól függően, egyszerre azonban nem alkalmasak több orientációból működni, továbbá forgatásra sem invariánsok.

Ide tartoznak azok az algoritmusok is, melyek szín alapján próbálják detektálni az arcot. „A szín-alapú megközelítés lényege, hogy minden pixelt osztályoz az alapján, hogy az árnyalata mennyire közelít az emberi arcszínhez, aztán pedig olyan területeket jelöl ki, ahol nagy tömegben találhatóak az emberi arc színéhez közelálló pixelek. A módszer előnye, hogy szemből és profilból is egyaránt jól működik, viszont nagyon érzékeny a bőrszínre és az arc alakjára” [Kövé05]. A szín alapú megközelítés további hátránya, hogy megvilágításra érzékeny, és önmagában nem elegendő a detektáláshoz. Előnye viszont, hogy forgatásra, eltolásra, átskálázásra invariáns.

1.2 Arcfelismerő technikák

„A kezdetek kezdetén az arcfelismerés a mesterséges intelligencia egyik legnehezebb részterületének tűnt. Az utóbbi évtizedben azonban rengeteg kutatási eredmény született, ami lehetővé tette a gyakorlati alkalmazásokat. A sikerek és a smart környezetek álma pedig mind a kutatók mind a támogatók figyelmét a témára irányította.” [Madai 12]

Geometriai tulajdonságokon alapuló módszerek

1973-ban Kanade által leírt módszer, ami a jellemző pontok közti távolságok arányaival dolgozott (szemek, száj, orr távolsága), egy 20 fős adatbázisában 45–75%-os felismerési aránnyal működött.[Kanade73]

1993-ban Brunellinek és Poggionak az orr hosszát, a száj elhelyezkedését, stb. használva 90%-os eredményt sikerült elérni egy 47 fős adatbázisban. Azt is kimutatták, hogy mintaillesztéssel 100%-os arányt is el lehet érni ebben az adatbázisban.[Brunelli93]

1995-ben Cox, Ghosn és Yianilos bemutatott egy több távolságon alapuló technikát, mely 95% megbízhatósággal működött 95 tesztképpel 685 fős adatbázisban. Azonban a távolságok meghatározására emberi közreműködésre is szükség volt. A technika gyenge pontja, hogy nem elég pontos az automatikus távolság meghatározás, pedig az eredményesség erősen függ ettől.[Cox96]

Sajátarc módszer

A módszer alapja hogy rendelkezünk egy képi adatbázissal, melyekben arcok szerepelnek, ezekből a képekből képzünk egy átlagot, és az átlagtól való eltérések alapján azonosítjuk az arcokat.

Az első ilyen rendszert 1991-ben Turk és Pentland mutatta be, amellyel az arcokról készült képeket az eredeti képek lényeges részeire vetítették. Ezek a lényeges részek a képek kovariancia mátrixának sajátvektoraiként határozhatók meg. A sajátvektorok képviselik az arcok különbözőségeinek jellemzőit. Azonban nem az összes vektort használják, hanem csak a néhány legnagyobb sajátértékűt, mivel ezek adják a legfontosabb jellemzőket. Új kép felvételekor a képből kivonják az összes kép átlagát. A különbség-kép, mint vektor és a sajátvektorok szorzatát képzik, így kapnak egy vektort, ami megadja az arc helyét a sajátvektor-térben. Annak érdekében, hogy a haj és a háttér változásait kiküszöböljék, csak az arc középső részét szokták használni a felismeréshez. Ez a módszer elég egyszerű és gyors, azonban érzékeny az intenzitás különbségekre, ezért a feldolgozás előtt érdemes a képeket normalizálni.[Turk91]

Sablon alapú illesztés

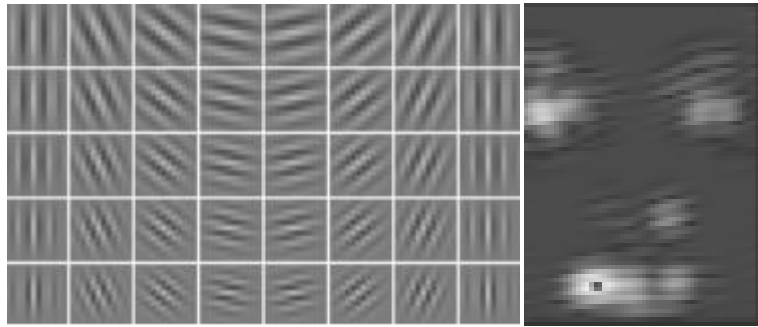
Az eljárás lényege hogy egy képet és egy sablon maszkot hasonlítanak össze és a különbséget eltárolják. Brunelli és Poggio megoldása a szemet, szájat, orrot és a szemöldök alatti arcot tárolja egy adatbázisban, majd az ismeretlen arcot az itt eltárolt arcokkal hasonlítja össze. Minden összehasonlítás egy különbség vektort eredményez, és amelyik vektor nagysága a legkisebb, az hasonlít leginkább a vizsgált arcra az eltárolt adatbázisban.[Brunelli93b]

Neurális hálózat alapú modell

A neurális hálózatok jól alkalmazhatók arcfelismerési és hasonló problémákhoz, de általában kis adatbázissal dolgoznak.



5. ábra 2D Gabor wavelet, balra a képtartományban, jobbra a frekvencia tartományban



6. ábra Balra: Gabor waveletek felülnézetből, jobbra a szűrés eredménye arcon

DeMers és Cottrell a legfőbb 50 tulajdonságot határoztak meg, melyet aztán ötdimenziósra redukáltak, ezután a felismerést több szintű perceptronnal végezték. [Demers93]

Gabor szűrők

Bizonyos arcfelismerő rendszerek speciális jellemzőket keresnek az arcon, és a teljes arc helyett csak ezeket tárolják, ezek alapján végzik az azonosítást. A különböző arcra illeszkedő gráfok létrehozásához is ilyen jellemző pontokra van szükség. A kétdimenziós Gabor szűrők (5. és 6. ábra) kiemelik a hegyeket, és völgyeket, amik arc esetén a szemek, száj, orr és egyéb jellemzők, kiemelkedő helyek (6.ábra). Ezáltal létrehozhatunk egy az arcra jellemző tulajdonság halmazt, amely egyedi és személyenként változó. A módszer lényege hogy több, különböző irányú és frekvenciájú Gabor szűrőket közelítő maszkokkal végig megyünk az arcon, és ezek a szűrők kiemelik a jellemző pontokat. Ahol jellemző pontot találtunk, ott eltároljuk az összes maszk értékét, így egy összes maszk hosszúságú vektort kapunk, ezeket eltároljuk, illetve felismerésnél ezeket a vektorokat hasonlítjuk a kérdéses arc vektoraival.

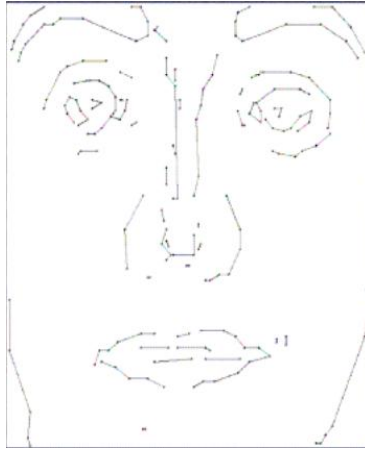
Háromdimenziós technikák

Az ilyen módszerek lényege, hogy a detektált arcból, egy 3 dimenziós modellt képeznek, és azonosításnál ezeket a modelleket hasonlítják össze. Ez a módszer több kamerás rendszert és nagy teljesítmény igényel.

Edge mapping

[Takács98]

Ez a technika először egy élkeresést hajt végre az adatbázisban lévő képeken, majd az így kapott gradiens nagyságát tároló mátrixokban binarizál. Ekkor rendelkezünk minden adatbázisban lévő képünkhöz egy-egy az élpixelet tartalmazó képpel (7. ábra). Azonosításkor, a vizsgált képen ugyanezeket a lépéseket hajtja végre az algoritmus, majd összeveti az adatbázisban tárolt képekkel, mégpedig a következő módon: minden, az éleket tartalmazó képet halmaznak tekint, és ezeknek a halmazoknak veszi a Hausdorff –



7. ábra Az éleket tartalmazó kép

távolságát. A módszer elég gyors, és az irodalomban található eredmények alapján, megbízhatóan működik.

Line Edge Mapping

Ez a technika a korábban említett Edge Mappingen alapul, azon ötlettel továbbfejlesztve, miszerint több információt tartalmaz, ha az éleket, mint összefüggő egyeneseket tekintjük. Ezzel az elgondolással, kevesebb adatot kell tárolnunk, hisz az egyenes szakaszoknak elegendő csak a két végét nyilvántartani. A következő lépésben, a már így meghatározott egyenesek között számolja a távolságot, és ezek alapján határozza meg a halmazok közti Hausdorff-távolságot.

2. Hasonló rendszerek

Saját rendszerem kifejlesztéséhez nemcsak az irodalomban megismert technikákat szeretném felhasználni, hanem az arcdetektálás és arcfelismerés területén elkészített rendszerekből levonható következtetéseket is alkalmazni szándékozom. Ezért ebben a fejezetben ilyen alkalmazásokat mutatok be, majd értékelem azokat.

Cerberus

[Bors03]

Ez a projekt egy komplexebb biometrikus azonosító rendszert valósít meg. Foglalkozik hangfelismeréssel és arcfelismeréssel is, mivel én csak arcfelismerést szándékozok megvalósítani, ezért csak azt ismertetem. Az arcdetektálását a következő lépések segítségével végzik:

1. Szürkeárnyalattá konvertálás
2. Medián szűrő - pontszerű zajok eltávolításához
3. Kontúrszűrő - az élek kiemeléséhez
4. Binarizálás – a későbbi kereső algoritmus bináris képpel dolgozik
5. A binarizált képen Randomized Hough Transform segítségével ellipszist keresnek
6. Az eredeti kép YCbCr színtérbe konvertálása –sok algoritmus használja mivel megfigyelések szerint a szem körül magas Cb és alacsony Cr értékek, illetve száj körül magas Cr értékek vannak, ezért könnyen lehet következtetni ezek elhelyezkedésére
7. Erózió – a szemek és száj kiemeléséhez.

A keresés alapja, hogy az arcot, mint ellipszis tekintik, majd ha sikerült egy ellipszist találniuk a képen, akkor ellenőrzik, hogy az tartalmaz-e arcot is. Az azonosítás Gabor szűrőkkel történik. 5*8 darab Gabor szűrőt alkalmaznak, 8 különböző szögben, és 5 különböző fázisban, és az ezek által kapott képeket hasonlítják össze. A Cerberus gyenge pontja a szem, de főleg a szájdetektálás pontossága, ha azonban ez sikeres, akkor kielégítő eredményt produkál a program. A tesztjeikből látható, hogy a legtöbb időt az ellipsziskeresés viszi el, ám még így is 2 másodperc alatt kiértékel egy képet (mai hardverekkel ez akár 1 másodperc körül is lehet).

Az általuk arcfelismeréshez használt Gabor szűrős technika megfelelő teszt eredményeket produkál, viszont nem képes valós idejű feldolgozásra, így az én dolgozatom céljainak nem felel meg.

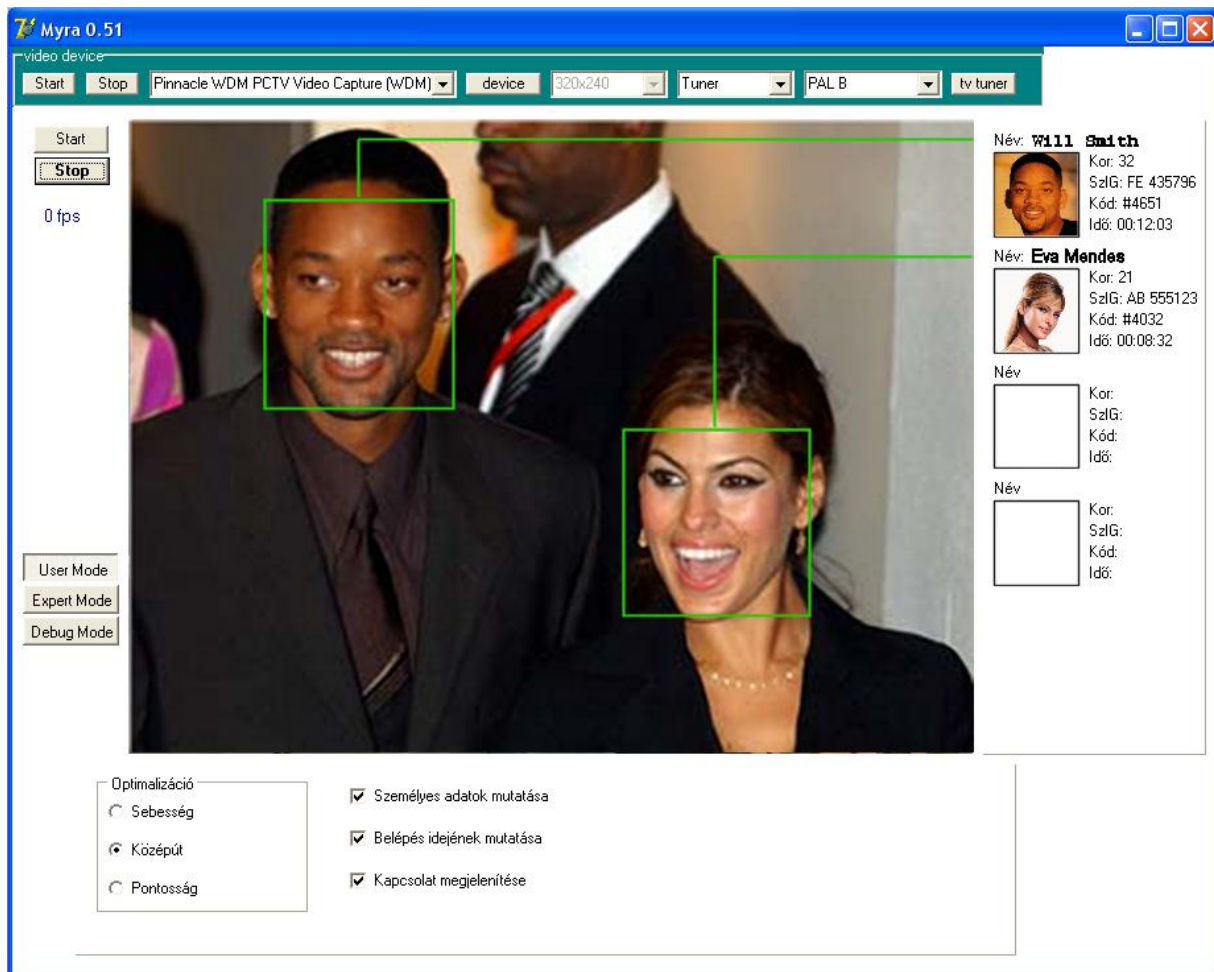
Myra

[Kövé05]

Ez a projekt az arcdetektáláshoz egyaránt használja a Viola-Jones és a bőrszín-keresés eljárásokat is. Először Viola-Jones módszerrel keresnek arcot a képen, ha azzal találtak egy területet, akkor ellenőrzik, hogy elegendő mennyiségű bőrszínű pixel található-e az adott területen. Ha igen, akkor elfogadják, mint találat, majd ezt próbálják követni. A követés úgy valósítják meg, hogy figyelik az elmozdult bőrszínű pixeleket a korábban detektált arc egy környezetében. Így gyorsul a detektálás, hisz csak akkor kell újra az egész képet vizsgálni, ha az arc „elveszett”.

Az arcfelismeréshez több módszert is implementáltak, de ezek által adott eredményeket, nem összesítették. Elkészítették a sajátarc, és a Gabor waveletes módszereket. Az Eigenface

algoritmusuk hatékonyan működik (8. ábra). A Gabor szűrős moduljuk nem nyújt ilyen megbízható eredményt, továbbá jelentősen lassabb, mint a sajátarcok módszere.



8. ábra Myra működés közben. Eigenface módszer által kapott eredmény.

Face Detection

[Kim04]

Ebben a projektben csoportképeken való arcdetektálást tűzték ki célul. A módszer alapja a szín alapú detektálás. Ahhoz, hogy valamelyest ellensúlyozzák a megvilágításból adódó különbségeket, nem RGB színtérben dolgoznak, hanem itt is YCbCr színtérbe térnek át. Itt jól meg tudják különböztetni a bőr és nem-bőr színeket. Áttérés után egy megfelelő küszöbértékkel binarizálják a képet, így ahol bőrszín találtak, ott fehér lesz, máshol fekete. A kamera zajából adódó mérési hibák miatt, illetve, hogy a szem körül sötétebb képet kapunk, ezért a binarizált képen eróziót alkalmaznak, így az arc közbelső részeiről eltűnnek az esetleges fekete foltok. Viszont mivel alapvetően csoportképeken teszteltek, így az egymás mellett álló arcok „összeolvadhattak”, tehát ezeket külön kell választani valahogy. Ezért egy élkeresést hajtottak végre az eredeti képen, és az így kapott bináris képet, ahol feketével jelölik az éleket, összeadják a korábban kapottal, ezáltal az összeolvadt, de valójában külön arcok szétválnak, külön szegmenseket alkotnak. Ezeknek a szegmenseknek meghatározható a középpontja, kivágható a téglalap alakú arcrégió. Mivel az szegmentálás során előfordulhatott, hogy egy arcról kisebb részeket levágott az algoritmus, és ezeket külön objektumként kezeli, ezért szükséges egy bizonyos „összeillesztő” lépés. A projekt készítői az egyszerű box-merge algoritmust alkalmazták. A nem-arc találatok kiszűrésénél, abból



9. ábra Működés közben a detektáló rendszer

indultak ki, hogy a vizsgált csoportképeknél az arcok közel azonos térségben helyezkednek el (9. ábra), míg például a karok ettől messze. Így korreláció segítségével, megállapítottak egy átlag egyenest, és az ettől való eltérést vizsgálva, döntötték el, hogy az adott objektum arc-e.

Tesztjeik alkalmával, olyan képekkel dolgoztak, melyeken 21-25 arc volt. Ezeknél a képeknél átlagosan 2 arcot nem talált meg, és egy olyan találat volt, ami nem arc. Egyszer sem fordult elő, hogy egy arcot többször azonosított volna. Futási időt tekintve 90-110ms alatt futott le az algoritmusuk képenként egy P3-as 700Mhz-es processzorral szerelt számítógépen.

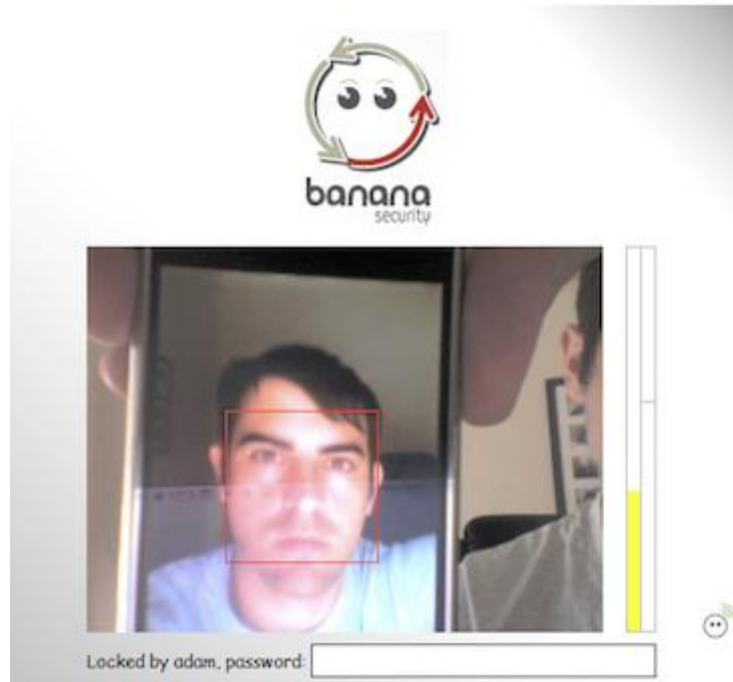
Összességében sikeresnek mondható ez a projekt, a futási idő, és az eredmények meggyőzőek, viszont a feladat túlzottan speciális, kis részben fedti az én feladatomat. Azt sikeresen bemutatták, hogy szín szelekcióval is lehet jól működő arcdetektálót készíteni.

Bananascreen

[Banana12]

Ez egy kereskedelmi rendszer, célját tekintve is hasonló az én projektem tervéhez. A működése egyszerű: telepítés után el kell menteni a kamera képét, amin a jogosult személy arca is szerepel majd újraindítás után, a belépésnél, ha felismer a program, akkor automatikusan beléptet, különben pedig csak akkor tudunk bejelentkezni, ha tudjuk a felhasználó nevét és jelszavát is. Összességében tehát ez egy kényelmi funkciókat nyújtó program.

Felhasználók arra panaszkodtak, hogy könnyen becsapható egy fényképpel, vagy mint az páran írták, úgy játszották ki a rendszert, hogy rá irányították a kamerát egy belépésre jogosult személyre miközben az nem figyelt.



10. ábra Bananascreen működés közben (a felhasználó épp megpróbálja becsapni a programot)

Jól látható (10. ábra), hogy megtartották felhasználó név- jelszó belépési lehetőséget is, hisz nem 100%-os az arcfelismerés, így előfordulhat, hogy jogosult személyt kizár a rendszer, ez pedig nem megengedett. Továbbá fel kell készíteni arra is, érzékeljen bizonyos interakciókat, hogy ne lehessen egy fényképpel kijátszani.

Értékelés

Mint látható hasonló, arcfelismerésen alapuló beléptető rendszerek már léteznek, mint amelyet én szeretnék alkotni, bár a kívánt biztonsági szintet egyik sem éri el véleményem szerint. Az én megközelítemben ez nem egy kényelmi funkció lenne, tehát nem felváltaná a jelszavas belépést, hanem kiegészítené azt. Mivel gondolni kell arra is, hogy nem 100%-os az arcfelismerés és elég kellemetlen lenne, ha a jogosult személy nem tudna belépni valami technikai probléma miatt, ezért szükséges lehet egy mesterjelszó, amit akkor lehet használni, ha a „normál” felhasználónév/jelszó megfelelő volt, de az arcfelismerés valamiért nem volt sikeres. Fontos szempont továbbá a sebesség is, mivel senki sem szeret sokat várakozni a belépésnél. Profilból való felismeréssel nem kívánok foglalkozni, hisz ez szerintem nem a feladat része (aki éppen belép a rendszerbe, az szembe néz a monitorral), és adott esetben csökkentheti is biztonságot. Speciális esetként le kell kezelni az a helyzetet, amikor nincs arc a kamera képen (vagy illetéktelen behatolás miatt, vagy csak nem működik megfelelően a kamera), illetve ha több arc is megjelenik.

3. Saját rendszer bemutatása

3.1 Célpontosítás

Arcfelismerés során fel kell tudnunk ismerni azon genetikai különbségeket, melyek jól megkülönböztethetővé tesznek minket, és melyeket számítógéppel is jól érzékelhetünk.

Mint már korábban írtam, problémát okoz majd a kamera képének minősége, ami származhat zajból, vagy az optika gyengeségéből. További problémaforrás lehet maga az ember, a mimikájuk, a bőrszín változása (smink, sérülés), különböző „kiegészítők” (piercing, tetoválás), öregedéssel járó deformációk (ráncok), és az ikrek megkülönböztetése is probléma. Nem szabad figyelmen kívül hagyni a környezeti tényezőket se, mint például a megvilágítást, árnyékokat. Ezen problémák kiküszöbölésére különféle előfeldolgozásokat kell majd használni, ami a lényegtelen információkat, zajokat eltünteti, a lényegeseket kiemeli.

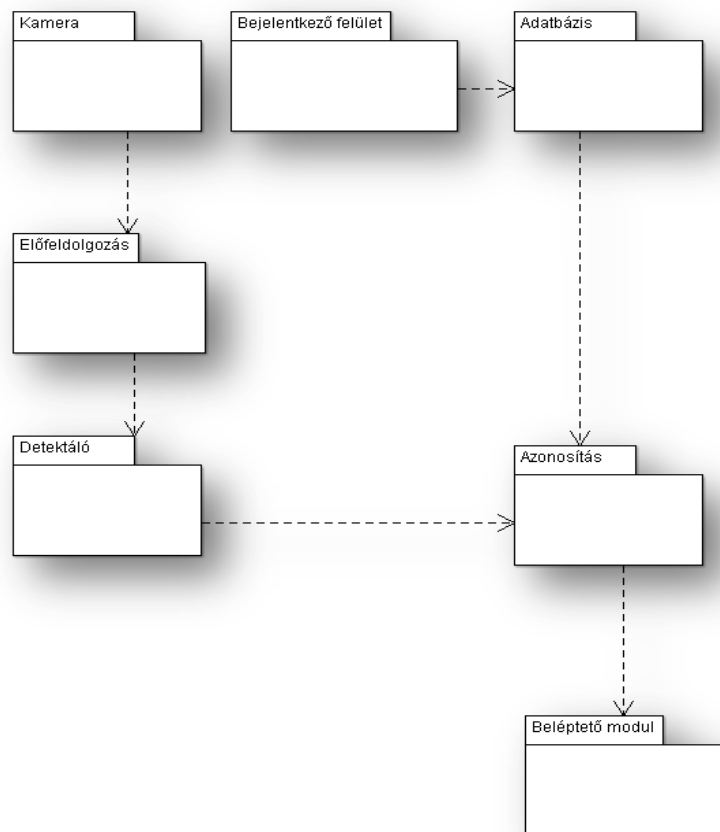
Tehát a cél egy olyan rendszer megvalósítása, ami:

- képes eldönteni, hogy van-e vizsgálandó objektum (arc) a képen, és ha igen, akkor meg kell vizsgálni, hogy pontosan egy arc van-e, ha nem, akkor el kell dönteni, hogy melyiket azonosítsuk
- észreveszi a genetikai különbségeket, az azonosítási pontokat
- mivel többszörös azonosítást tervezek megvalósítani, ezért az algoritmusok egy összesítése lesz a kimenet. A nagyobb pontossággal működő algoritmusok valószínűsíthetően nagyobb súlyozást kapnak

Mivel ez egy beléptető rendszer lesz, kulcsfontosságú kérdés a sebesség, tehát a felhasznált algoritmusok, technikák is e szerint kerülnek kiválasztásra. A feladat alapvetően nem az azonosítás, hanem a verifikálás, azaz a felhasználó által megadott fiókhoz való hozzáférés jogosultságának ellenőrzése.

3.2 Rendszervázlat

A rendszer hét modulból fog állni (11. ábra) ezek közül csak a detektáláshoz és felismeréshez szükségesek kerülnek bemutatásra. Alapvetően kameraképpel dolgozok, ahonnan a képfolyam egy előfeldolgozóba kerül. Ez a különböző algoritmusok számára előkészíti a képet. Innen az aktuális képkocka tovább megy a detektálóhoz, és az azonosító modulhoz is. A detektáló modul megkeresi a képen az arcot, ha több arc van a képen, akkor azt, amiről a legvalószínűbb, hogy keressük (a legnagyobbat), és ennek a koordinátáit tovább adjuk az azonosítónak. Az azonosító modul megkapja a képet, és a felhasználó adatait is. Ezek után kapcsolódik az adatbázishoz, ahonnan lekérdezi az felhasználótól kapott adatokhoz tartozó információt, képeket. Ha a felhasználóhoz egyáltalán nem talál adatot, akkor az azt jelenti, hogy eleve jogosulatlan belépési kísérlet történt. Ekkor leállítjuk a feldolgozást, és várunk a következő belépési próbálkozásig. Ha találtunk az adatbázisban rekordot, akkor elkezdjük az azonosítást. Az azonosító hoz egy döntést, miszerint a felhasználó vagy jogosult belépni, vagy nem. Ha azt a döntést kapjuk, hogy nem, folytatjuk a kamera kép vizsgálatát, amíg nem kapunk olyan képet, amiről pozitív döntést kapnánk, vagy a felhasználó be nem írja a „mesterjelszót” (erre azért van szükségünk mivel a helyes működés nem garantált, ezért fals-pozitív döntések is születhetnek). Ha pozitív döntést kapunk, akkor engedélyezzük a rendszer használatát, és a program leállhat.

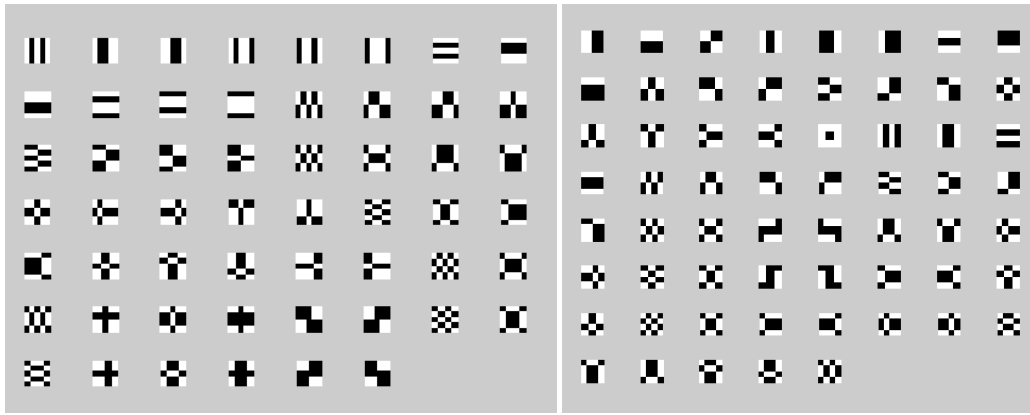


11. ábra A rendszer csomagdiagramja

3.3 Modulok részletes kifejtése

Előfeldolgozás

Minden képfeldolgozó program esetén szükség van valamiféle előfeldolgozásra. Mivel a projekt kameraképpel dolgozik, ezért ebben az esetben ez hatványozottan érvényes. A választott algoritmusok nem veszik figyelembe a színek különbségeit, így szürkeárnyaltos képen történnek a kiértékelések, ezért első lépésben egy szürkeárnyaltos átalakításra van szükség. Ezután szétválik a folyamat, hisz a különböző felismerő algoritmusoknak, más-más igényük van. A felismeréshez használt Eigenface módszer nagyon érzékeny a megvilágítás változására, ezért ezt előfeldolgozás során kompenzálni kell. Az ötlet az, hogy vizsgáljuk meg az arcként detektált képrészlet átlag intenzitását, ha túl kicsi, akkor növeljük a fényerőt, ha túl nagy, akkor csökkentjük, és ezt lehetőleg folyamatos skálázással. Ezt gamma korrekcióval valósítom meg, melynek paraméterét egy egyszerű lineáris skálázás során kapom meg. A skálázáskor használt referenciaszám megválasztását, a teszteleseknél részletezem. Ez a módszer jól működik, ha a globálisan változik a megvilágítás színe, viszont a különböző irányokból történő bevilágításokra, nem jelent megoldást. A másik, felismerésre használt algoritmus, az Edge Mapping az éleket tartalmazó bináris



12. ábra Az összes (115 darab) Haar szűrő maszkja

képpel dolgozik, így ezt is elő kell állítani. Mivel később szándékozok implementálni egy másik, szintén éllel dolgozó algoritmust, ezért célszerűnek tűnik ezt csak egyszer, az előfeldolgozás során megvalósítani. Első lépésben az arcként detektált területen egy Gauss szűrést hajtok végre, így a „gyenge” éllel elmosódnak, és csak az „erős”, fontos, valódi éllel maradnak felismerhetők a képen. Ezután az éllel detektáláshoz a Canny éldetektort használom, mivel ez kellőképpen gyors, az éllel elvékonyítja egy pixel szélességre, és amely éllel feltehetően összetartoznak, azokat összeköti.

Az arcdetektálás

[Viola04]

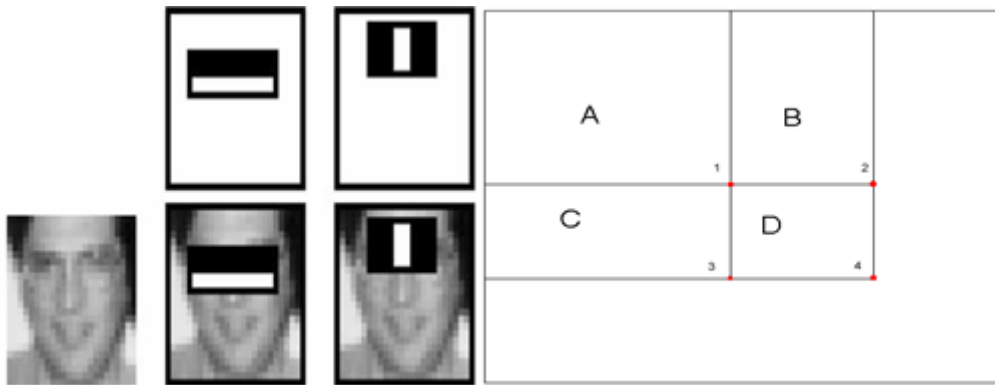
Az arcdetektálás során a Viola-Jones féle objektum detektort használom. A módszerük univerzális, bármilyen objektumra használható, így tehát arcra is. Az algoritmus a 12. ábrán látható Haar maszkok közül használ néhány kiválasztottat az arcok megkereséséhez. A maszkok kiválasztása a betanítástól függ. A kiválasztott maszkokkal kell megvizsgálni a képet (13. ábra), hogy meghatározzuk, van-e arc a képen. Mivel kezdetben rengeteg jellemző tartozik egy területhez (nagyságrendekkel több, mint ahány pixelt a terület tartalmaz), ezért ezek közül ki kell választani azokat a jellemzőket, amik valóban jól el tudják dönteni, hogy az adott terület archoz tartozik-e. Erre az AdaBoost algoritmust alkalmazzák. Az algoritmus lényege, hogy alacsony szintű osztályozókat használnak, egy-egy kapcsolatot létrehozva egy osztályozó és egy jellemző között, és megvizsgálták negatív és pozitív példákra mely jellemzők a leghatásosabbak. Az algoritmus gyorsaságát az biztosítja, hogy az összes Haar jellemző téglalapokra bontható, így ezek számítása igen gyors.

Ehhez szükség van a vizsgált kép integrális képére. Ezt a következő képlettel lehet kiszámolni:

$$ii(x, y) = \sum_{x' < x, y' < y} i(x', y')$$

ahol $ii(x, y)$ az integrális kép (x, y) pozíciójában lévő érték, $i(x', y')$ pedig az eredeti kép (x', y') pozíciójában lévő érték. Ezután a jellemzőkben látható téglalapok számolása csupán tömbindexelési feladat, és 1-1 kivonás, illetve összeadás. A D téglalap számítása a 13. ábrán például a következő módszerrel határozható meg:

$$D = ii(4) - ii(3) - ii(2) + ii(1)$$



13. ábra Maszkok működése (balra), és az integráliskép (jobbra)

Végül pedig, ami tovább gyorsítja a folyamatot, hogy a jellemzőket kaszkád struktúrába rendezték, így ha valamelyik jellemző szerint nem lehet arc a területen, akkor a vizsgálat azon a részen nem is folytatódik, ha pozitív visszajelzést kapunk, akkor folytatjuk a következő jellemzővel, és így tovább egészen addig, amíg nem érünk az utolsó jellemzőhöz, vagy nem kaptunk negatív visszajelzést. Ez nagyban hozzájárul a sebesség, és a pontosság növeléséhez is, hisz ha már az első jellemző szerint sem lehet arc, akkor nem foglalkozunk azzal a területtel tovább, és csak azon részeket fogadjuk el, amelyek az összes jellemző szerint arc.

A megvalósítás az OpenCV algoritmus gyűjtemény segítségével történik, amiben implementálva van a Haar objektumkereső, így megfelelő kaszkád láncsal „bármit” kereshetünk vele. A kereső függvény egyik lényeges paramétere a pásztázó maszkok kezdő méretei. Ha ezeket túl kicsire választjuk meg, akkor az algoritmus lassan fog végre hajtódni, ha túl nagyra, akkor a megadott méretnél kisebb arcokat nem fogja megtalálni. További fontos paraméter a skálázás, hogy milyen lépésközökkel növelje az ablakok méreteit, ha ezt kicsire választjuk, szintén lassul a program, míg ha túl nagy, akkor sok lehet a felesleges (háttér) terület a detektált részen. Az OpenCV tartalmaz alapkaskádokat, arckereséshez, ám ezek találati aránya nem megfelelő, ha nem pontos az arc orientációja. Így jelenleg a korábban már említett Myra projektben is használt kaszkádot használok, aminek a működése kielégítő eredményt ad, azonban jelentősen lassabb. Hogy ezt a sebesség lassulást ellensúlyozzam, ezért miután megtaláltam egy képkockán az arcot, annak a pozícióját eltárolom, és a következő képkockánál, csak annak egy környezetét vizsgálom először. Ezt a környezetet ROI-nak (Rectangle Of Interest) nevezik. Ha ott nem találok arcot, akkor az egész képet vizsgálom. A követő ablak méretének megválasztása felhasználói szokások elemzése alapján történt. Az itt említett paraméterek tesztek alapján kerülnek beállításra.

Arcfelismerés

A projekt több algoritmust is használ felismerés céljából. Ezek egymás mellett párhuzamosan működnek, és ezek súlyozott eredménye adja majd a végső döntést. Az egyik használt módszer a főkomponens analízisen alapuló módszer, azaz a sajátarc technika. Egy másik módszer az EM (Edge Mapping), ami az arcon található éleket hasonlítja össze. Természetesen az összes módszernek szüksége van egy arc adatbázisra (14. ábra), amiben eltároltához hasonlíthatja a vizsgálandó arcot. Ebben az adatbázisban különböző fényviszonyoknál készült, különböző orientációjú és akár különböző kamerával felvett képek is lehetnek.



14. ábra -Adatbázisban tárolt arcok, több féle kamerából, és különböző fényviszonyoknál



15. ábra Az átlagarc

Saját arcok

A saját arcok algoritmus egy statisztikai módszeren, a főkomponens analízisen alapul. Ennek során egy többváltozós rendszert elemeznek, és az elemzés során meghatározzák, hogy mely változók vannak nagy befolyással a rendszer működésére, és melyek hanyagolhatóak el. A módszer gépi látásnál történő felhasználásakor, a képeket egydimenziós oszlopvektorként kell kezelni. Fontos kritérium a képekkel kapcsolatban hogy ugyanolyan méretűek legyenek. A képek átalakítása után, egy T mátrixot hozunk létre. Ebben a mátrixban eltárolt képekből meghatározunk egy átlagképet egy egyszerű átlagszámítással (15. ábra). Következő lépésben kiszámoljuk minden kép eltérését az átlagtól, az így kapott különbségeket elmentjük egy A mátrixba, majd ezekből kiszámoljuk a T mátrix kovariancia mátrixát. A kovariancia mátrix sajátvektorait képpé alakítva láthatjuk a sajátarcokat (16. ábra). A kovariancia mátrix számítása eltér a szokásos főkomponens analízistől, mivel az egy $N^2 * N^2$ méretű mátrixot eredményezne, ezért helyette egy $L = A^T * A$ mátrixot számolunk, ami egy $M * M$ méretű mátrixot eredményez, ahol N a képek mérete, M pedig a képek száma. L mátrix sajátvektorainak lineáris kombinációjából előállítható Cov sajátvektorai. A következőképpen számítható ki Cov mátrix saját vektorai: $V_i = A * u_i$ ahol V a kovariancia mátrix sajátvektora, és u pedig az L mátrix sajátvektora. Az így kapott sajátvektorokkal, az arcok és átlagarc közti különbségeket végigszorozzuk, aminek eredményeként megkapjuk a sajátarcokat.



16. ábra Az adatbázisban tárolt sajátarcok

Az azonosítás úgy történik, hogy beolvassuk a vizsgálandó képet, ennek vesszük az átlagtól való különbségét (17. ábra), majd a különbséget beszorozzuk a már korábban kiszámolt bázisvektorokkal, és az így kapott mátrixot összehasonlítjuk az eltároltakkal. A legkisebb különbségűre hasonlít leginkább a kép (18. ábra). A módszernek két létfontosságú pontja van: az egyik a képek normalizálása: azonos méretre vágás, a megvilágításból adódó különbségek minimalizálása, a másik pedig a döntésnél a megfelelő küszöbérték megválasztása, hisz nem elég a legközelebbi értéket elfogadni, meg kell nézni, hogy nem túl nagy-e a különbség.

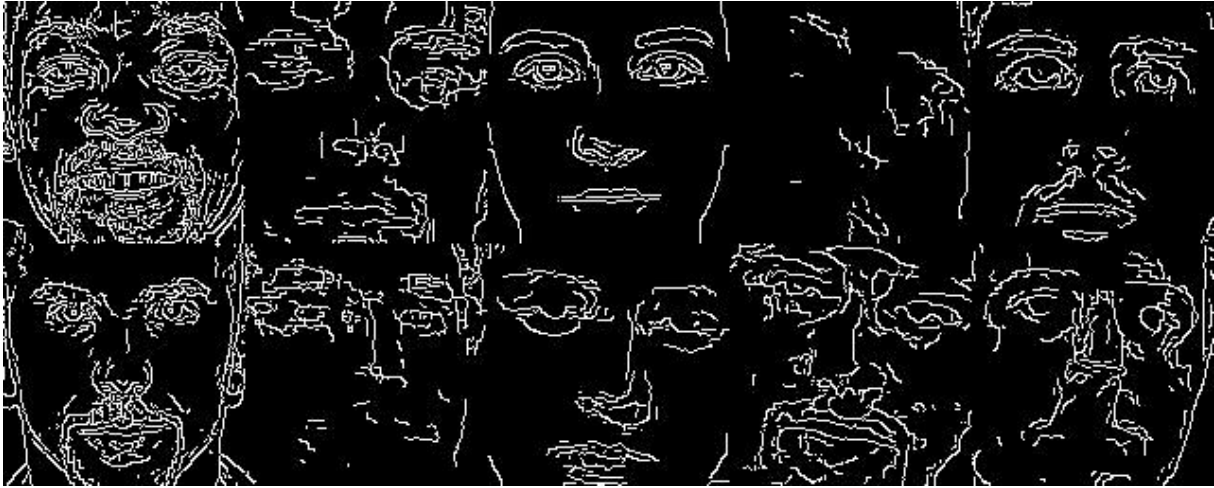
E módszer 10-20 fős adatbázis esetén működik a legjobban, ennél kisebb adatbázisoknál nincs elég tanulóminta, így a kiválasztott vektorok nem biztos, hogy elég reprezentatívak, míg nagyobb adatbázis esetén, már kevésnek bizonyulhat a kiválasztott vektorok száma, ha pedig növeljük a vektorok számát, akkor lassul az algoritmus.



17. ábra Vizsgált kép, és abból készült sajátarc

6759	6732	3758	5232	2875
4986	5028	4139	8155	6558

18. ábra Az adatbázis képeitől mért távolságok



19. ábra Az adatbázisban tárolt képek éltérképei

Edge Mapping

A sajátarc módszer mellett az *Edge Mapping* algoritmust valósítottam meg a felismerés rész hatékonyságának javítása céljából. Ez a módszer is csak az arcot tartalmazó képrészletet használja, a háttér csak zavarhatja a működésben. Első lépésben valamely éldetektorral meg kell állapítani az arc szignifikáns éleit, ezekből egy bináris képet kell alkotni. Ahhoz, hogy a jelentéktelen éleket eltűnjenek, az élkeresés előtt egy Gauss szűrést végzek a képen. Ez eléggé elmossa a képet, akár csak az átlagoló szűrő, viszont azzal szemben, itt nem jelennek meg begyűrűzések. Miután a jelentéktelen élek eltűntek a képről, az élkeresést a Canny éldetektorral végzem el, mivel az algoritmus által előállított bináris képen az élek csak egy pixel szélesek, és a lehetséges összetartozó élek össze vannak kötve. Az így előálló bináris képek (19. ábra) kerülnek tárolásra. Azonosításkor ugyanezeket a detektorokat és értékeket alkalmazva a vizsgált arcra (20. ábra) is készíteni kell egy ilyen éleket tartalmazó képet, majd ezt kell összehasonlítani a tároltakkal. Az összehasonlítás során a képeket halmazként tekintjük, melyek elemei az élpixekek, és ez alapján számoljuk ki a képek Hausdorff-távolságát:

$$h(I_1, I_2) = \frac{1}{|I_1|} * \sum_{i \in I_1} \min_{j \in I_2} ||i - j||_L$$

ahol I_1 és I_2 az éleket tartalmazó bináris képek, $|I_1|$ I_1 -nek a mérete, és $||i - j||$ i és j pontok távolsága. Tehát minden i ponthoz megkeressük a legközelebbi j pontot, ezeket a minimum távolságokat összegezzük, majd pedig elosztjuk $|I_1|$ értékkel. Az így kapott különbségeket összehasonlítjuk, és akár csak a sajátarcok módszerénél, itt is a legkisebb távolságút tekintjük találatnak, persze csak ha egy meghatározott küszöbérték alatt van.



20. ábra A vizsgált arc, és a hozzátartozó éltérkép

Döntés meghozása

Mivel több algoritmust is használok, ezért az ezektől kapott eredményeket összegezni szükséges. Az összegzés során figyelembe kell venni alapvető feladatot, vagyis azt, hogy a projekt célja egy verifikációs rendszer, tehát csak azt kell eldönteni, tényleg az a személy van-e a kamera képen, akinek a nevét a felhasználó megadta. Ha igen, akkor beléphet, ha nem, akkor folytatódik tovább a feldolgozás. Az összegzés úgy történik, miután összehasonlítottuk a detektált arcot az adatbázisban tárolttal (ha egyáltalán van ilyen megadott nevű személy az adatbázisban), akkor mindegyik algoritmus eldönti, hogy lehet-e az adott személy, akinek vallja magát. A később részletezett teszteredmények alapján, a két algoritmus eredménye egyenlő súllyal kerül beszámításra. A számítás menete a következő: sajátarcok módszere 0,3-at ad az eredményhez, ha az összehasonlított képek távolsága kisebb, mint a küszöbérték, és további 0,2-t, ha a minimális távolság, a megfelelő képnél érzékelhető. Edge mapping során ugyanezek a számok érvényesek, csak fordított esetben. Tehát 0,3-at a minimális távolsággal rendelkező képnek, és 0,2-t minden küszöbértéknél kisebb távolságúnak. Így minden eltárolt személyhez tartozik 0 és 1 között egy szám. Ha a bejelentkező felhasználóhoz tartozó szám 0,5-nél nagyobb akkor beléphet.

4. Tesztelések, eredmények bemutatása

Az előfeldolgozás eredményessége

Az előfeldolgozás során használt módszerek mind rendkívül gyorsak, a program sebességét szinte nem is befolyásolják. Előfeldolgozás során a képet átalakítom szürkeárnyalatossá. Mivel különböző megvilágításoknál fogják használni a programot, ezért fel kell készíteni erre. Ezt egy skálázás oldja meg, melynek referencia számának meghatározásához több képet kellett megvizsgálni. A tapasztalat azt mutatja, hogy az optimálisan megvilágított képek, átlag intenzitása 120-130 körüli. Ezen következtetés alapján a referencia számot 128-nak választottam meg, ezzel elosztva az aktuális kép átlag intenzitását, 0 és 2 közötti számot kapunk, amit a gamma korrekciónál használok fel. Ennek a korrigálásnak köszönhetően a felismerő algoritmusok megbízhatóan működnek különböző színű és intenzitású megvilágításoknál is. A korrekció után kapunk egy normalizált képet (21. ábra középső része), majd ezen végzem a detektálást és, a sajátarc alapú felismerést. Az éldetektálást is szürkeárnyalatos képen végzi a program, de előtte még egy Gauss szűrést hajt végre, egy 11*11-es maszkkal, ami az apróbb élszerűségeket elmossa, hogy azok a későbbiekben ne zavarják a felismerést. Ezután a Canny detektort 150-es élküszöbvel, és 80-as összekötési küszöbvel használok, így csak az elég erős éleket detektálja, utána viszont összekötésnél a gyengéket is beveszi, ha azok összetartoznak.

A detektálás eredményei

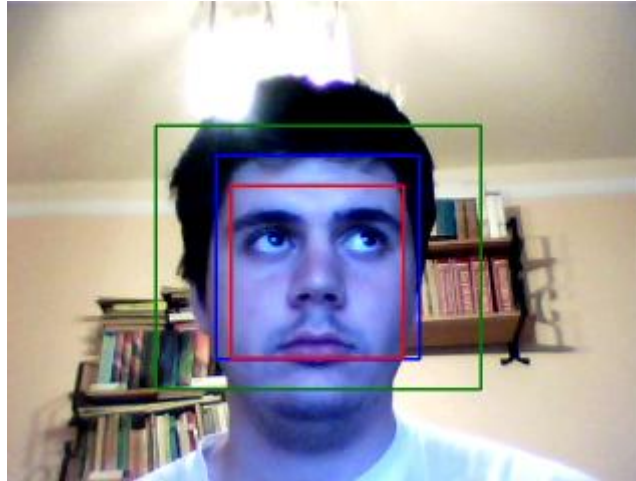
A Myra által használt kaszkád nagyon jó eredményeket produkál, az enyhén elforgatott arcokat is képes detektálni. Ez a működéshez tökéletesen elegendő, hisz feltételezem, hogy a program futása során a felhasználó a kamerába egyenesen néz. Azonban közel 50%-kal megnövekedett a futási ideje az OpenCV alapértelmezett kaszkád láncához képest, ezt a már korábban említett módszerrel, úgynevezett ROI használatával ellensúlyozom. Mivel ez egy beléptető rendszer, az arc méreteire vonatkozólag a következőket lehet feltételezni:

- a felhasználó nem fog a kamerától egy méternél távolabb tartózkodni
- a felhasználó nem ér a kamerához 30 centiméternél közelebb

Ezekkel a megszorításokkal mérések alapján az arc 250*250 (30 centiméteres távolság) és 60*60 (1 méteres távolság) méret között várható egy 640*480-as webkamera esetében. A detektáló paramétere ennek alapján nem lehet 60*60-as ablaknál nagyobb, hisz akkor nem találná meg az egy méterre lévő felhasználó arcát, tehát a vizsgáló maszkok kezdő mérete 60*60 pixel. Tapasztalatokon alapuló paraméterezés eredményéül azt kaptam, hogy a



21. ábra Az előfeldolgozás eredménye



22. ábra Az arc detektálása. Kékkel a detektált arc, zölddel a ROI, pirossal a leszűkített terület

skálázó faktort 1.3-ra érdemes állítani. Ezen a szinten elég gyors a detektálás, és a skálázás sem túl drasztikus. Egy másik a működést drasztikusan felgyorsító módszer a ROI. Ennek paraméterezése adja meg, hogy a detektált arcot tartalmazó téglalapot, mennyire terjesztem ki a következő képkocka vizsgálatakor. Mivel az arcméretek a kamerától való távolság függvényében elég nagy skálán mozoghatnak, így a kiterjesztés mértékét nem lehet fix pixelekkal megadni, a detektált arc méretétől függ. Ennek megállapítására videó felvételt készítettem a felhasználói szokásokról, és képkockaként elemeztem a detektált arc elmozdulásának értékét, az arc méretének függvényében. Ebből az látszódott, hogy horizontálisan az elmozdulás értéke intenzív, de még nem követhetetlenül gyors mozgás esetén 25% körül maximalizálódik. Ennél nagyobb elmozdulásoknál már annyira elmosódott az arc, hogy habár a detektálás sikeres volt, a felismerő algoritmusok akkor se adnának releváns eredményt. A vertikális mozgás ennél sokkal kisebb mértékű volt, itt olyan 10% körüli elmozdulások voltak a maximálisak. Némi „ráhagyást” biztosítva (mivel különböző kameráknál érzékelhető minimális különbség), a téglalapot horizontálisan 50%-kal, míg vertikálisan 30%-kal terjesztem (21. ábra) ki, így a detektálásnál, ha sikeres a ROI-val történő követés, akkor közel 5-10-szeres gyorsulás érhető el. A detektálás után nemcsak kiterjesztésre van szükség. Mivel a felismerésre használt algoritmusok nem kimondottan jellemzőket vizsgálnak, hanem a képet mint egészet tekintik és értékelik ki, ezért fontos, hogy a képen tényleg csak az arc jelenjen meg, ne legyenek felesleges, zavaró tényezők. Ilyen tényező a háttér, aminek jelenléte azonosítás során mindenképpen rontja az algoritmus eredményeit, de a haj szintén zavaró tényező lehet. Ezért meg kellett vizsgálnom a detektált arcokat, hogy azoknak mekkora részük a valóban lényeges. Ezt szintén egyenként, több képkockán kellett elvégezni, ám itt problémát jelentet az is, hogy mit tekintek hasznos résznek, és ezeket miként tudnám összehasonlítani. Az összehasonlítást végül manuálisan végeztem minden megvizsgált képkockán, így jelentősen kevesebb vizsgálatból (20 darab) hoztam meg a döntésem. Azért, hogy a mérés a kevés vizsgálat ellenére mégis reprezentatív legyen, ezért készült öt kép harminc centiméteres távolságból, öt darab egy méterről, a fennmaradó részt pedig a két véglet között véletlenszerűen próbáltam kiválasztani. A mérés azt mutatta, hogy a detektált terület szélességét összesen 15%-kal kell csökkenteni, méghozzá egyenlő részben, a magasságát szintén 15%-kal érdemes csökkenteni, viszont azt aszimmetrikusan. A detektált arcok alja szinte minden esetben az ajkak vonala, ha szimmetrikusan történne a csökkentés, akkor ezeket levágnák, pedig ez releváns adat, ellenben a homlokkal, ami könnyen összezavarhatja az algoritmusokat. A detektálás

Igaz pozitív	Igaz negatív	Hamis pozitív	Hamis negatív
76	15	5	4
95%	75%	25%	5%

23. ábra A detektálás eredményei

végeredménye a 22. ábrán látható, ahol be van jelölve a detektált arc (kék), a ROI (zöld), és a vizsgálandó terület (piros). A 23. ábrán látható a detektáló modul eredményessége. Az itt feltüntetett eredmények 100 mérést követően álltak elő, 80 esetben volt szembe néző arc a képen, míg 20 esetben vagy profilból volt látható, vagy egyáltalán nem tartalmazott arcot a kép.

A felismerés eredményei

A felismerés sikeressége, erősen függ a kamerától. Két, ugyanarról a személyről, akár ugyanolyan fényviszonyoknál, de különböző kamerával készített kép esetén az algoritmus nem várt eredményeket adhat. Még nagyobb mértékben befolyásolja a működést a megvilágítás is, ugyanis az előfeldolgozás során a különböző irányokból történő bevilágításokat nem szűri a program, és ez a felismerés során problémát okoz mindkét eljárás során.

A sajátarcok módszere egy-két emberes adatbázis esetén nem ad elég megbízható eredményt, 5-6 ember méretű adatbázis esetén már elég jó eredményeket ad, míg 10-20 főnél nagyon jó, ha „ideálisak” a körülmények. Ideális körülmény alatt azonos kamerával, hasonló megvilágítással készült képeket értünk, melyeken az arcok orientációja hasonló és a háttér nem látszódik. Kevésbé ideális esetben,

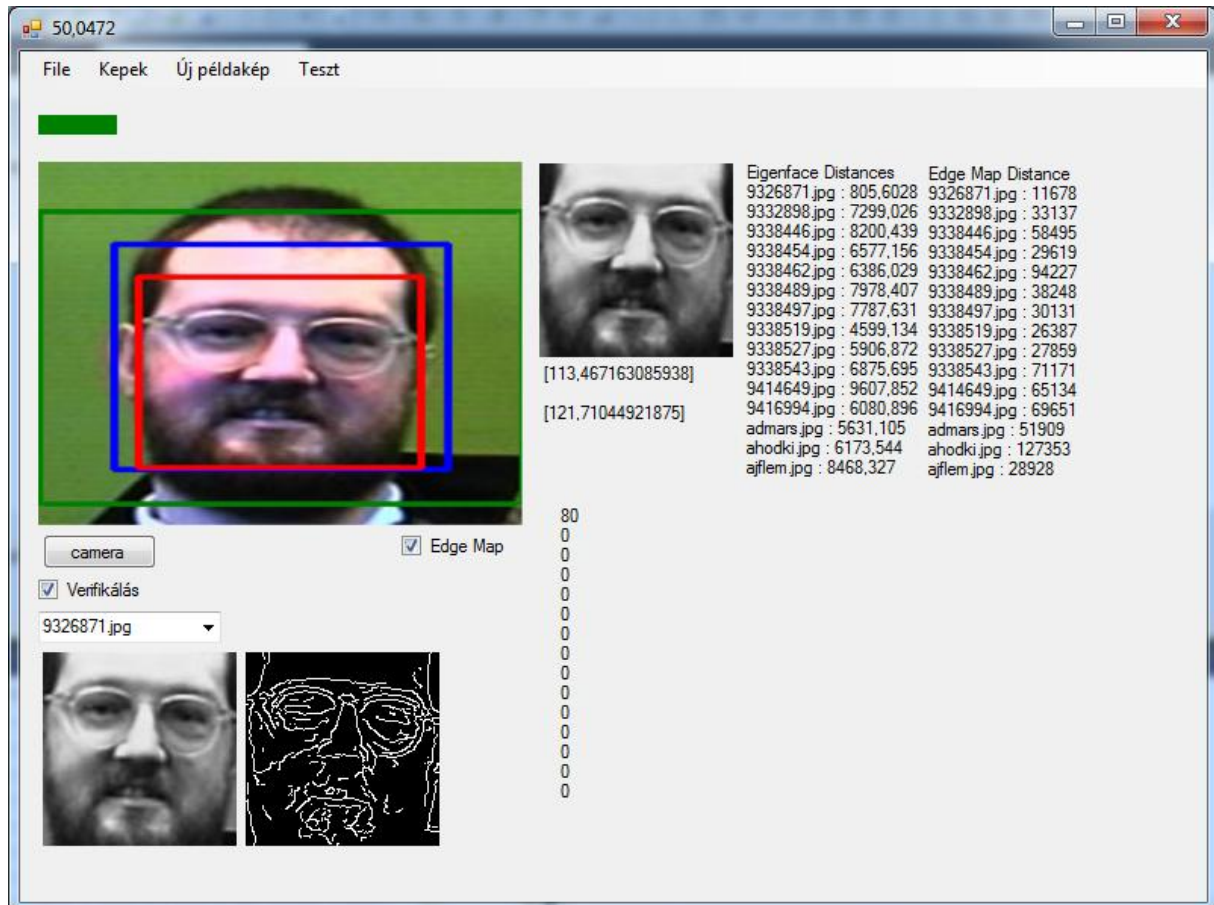
Az *Edge Mapping* algoritmus a globális megvilágítás változásaira elég érzéketlen, azonban az oldalról történő megvilágítás ezt az algoritmust is összezavarja, tipikusan az arc közepén egy új vertikális éleket hoz létre, ami megváltoztatja az eredményeket. Mivel itt a képek nincsenek egymásra kihatással, a „tanítás” során nem fontos az, hogy az orientációk egyezzenek, azonosításkor azonban komoly problémát tud okozni, ha jelentősen más orientációjú a vizsgált arc. További probléma, hogy mivel éleket detektál, erősen befolyásolja a szűrő az eredményt, hisz az is megjelenik a képen élként. Előnye viszont hogy átlagos megvilágításnál, azonos kameraképeket nézve jól megfelelően kontrolálja a saját arcok módszerét (24. ábra). Más kameraképek erősen megzavarják a működését, hisz rossz minőségű, kockásodó képeknél könnyen detektál olyan éleket, amik egy jobb kamera esetében nem is léteznek.

	Igaz pozitív	Igaz negatív	Hamis pozitív	Hamis negatív
Sajátarcok	73%	56%	44%	27%
Edge Mapping	2%	100%	0%	98%
Összegzés	86%	55%	45%	14%
Sajátarcok (oldalról bevilágítással)	53%	51%	49%	47%
Edge Mapping (oldalról bevilágítással)	3%	98%	2%	97%
Összegzés (oldalról bevilágítással)	40%	44%	56%	60%

24. ábra A felismerő modul eredményi

Az *Edge Mapping* algoritmus magas hamis negatív eredményeit az okozza, hogy nehéz megfelelő küszöb értéket találni, mivel a képek távolságai nagyon széles skálán mozognak,

visszont a döntés modul pontozásának köszönhetően a sajátarcok módszer erős kontrolljaként működik. A teszteket egy részét Dr. Libor Spacek [Spacek08] arc adatbázisán végeztem, az oldalról történő bevilágítás megfigyeléséhez pedig saját adatbázist használtam. Az eredményekből az látható hogy habár mindkét módszert megzavarja a bevilágítás, az *Edge Mapping* algoritmus kevésbé érzékeny rá, mint a saját arcok módszere. Tesztelés során megfigyelhető volt azonban, hogy a hamis pozitív esetek jelentős részénél mindkettő algoritmus határozottan felismerte a vizsgált arcot. A 25. ábrán látható a program futás közben, amikor is sikeresen detektál és felismer egy adatbázisban szereplő arcot. A bal felső sarokban lévő zöld négyzet jelzi a sikeres verifikálást.



25. ábra A program működés

5. Összegzés, jövőbeli tervek

A detektálásnál gyengének tűnhet, hogy csak szembe néző arcokat érzékeli az algoritmus, ám ez nem teljesen igaz, mert a jelenleg használt kaszkád lánc megenged egy kis elfordulást is tehát életszerűen működik. Nagyobb flexibilitást pedig felesleges lenne megengedni, hiszen akkor az a felismerésnél okozna problémát, és a sebesség rovására is menne. Fontos célkitűzés volt, hogy a program valós időben működjön. Ez részben sikerült hisz csupán a sajátarcok módszerét alkalmazva 20-30ms egy képkocka feldolgozásának ideje. Másrészről a sajátarcok kontroljaként alkalmazott *Edge Mapping* implementációja nem elég gyors, így ennek bekapcsolásakor drasztikusan csökken a sebesség. A rendszer jelenlegi formájával meg vagyok elégedve, a tesztesetek olyan széles körűek voltak, hogy ezekkel szemben megfelelően helytálltak az algoritmusok. Az egyes irányokból történő bevilágítás, viszont komoly akadályozó tényező.

A jövőben erre mindenképp kell megoldást találni, mivel mindkét algoritmust befolyásolja ez a jelenség, és ennek kiküszöbölésével sokat javulhatna a teljesítményük. Ezt a hibaforrást még, előfeldolgozás során lenne a legcélszerűbb kiküszöbölni. További fejlesztési lehetőségként tekintem az *Edge Mapping* módszer gyorsítását, hogy a sajátarcok módszere mellett ez is valós időben fusson. Ezzel kapcsolatban előtérbe kerül a GPU nyújtotta számítási kapacitás kihasználása is, mely jelentősen tudná fokozni az algoritmusok sebességét. Mindezek mellett talán szintén fontos feladat újabb algoritmusok implementálása, olyan vagy olyanoké, melyek ellensúlyozhatják a már meglévők gyengeségeit és nincs túl nagy számítás igényük. A *Line Edge Mapping* módszer az irodalom szerint nagyon jó eredményeket produkál, az oldalról történő bevilágításra is invariáns viszont lassabb az *Edge Mappingnél* ezért, nem kerül implementálásra.

További fejlesztési lehetőség még a döntések skálázása, hogy ne csak konstans értékeket a egyes képeknek. Továbbá a meglévő rendszerek küszöbértékeinek megválasztását is érdemes lehet finomítani.

Ezekkel a módosításokkal véleményem szerint, sikerülne elérni a 80% megbízhatóságot, ami után a felhasználói interfész véglegesítését, és az operációs rendszerbe való integrálást szükséges elvégezni.

Az elkészült rendszer úgy vélem teljesíti a vele szemben támasztott követelményeket. Sikerült csupán holisztikus, kis számításigényű módszerek használatával, egy stabil, 71%-s megbízhatósággal működő arcfelismerő rendszert létrehozni. Mindkét felismerésre használt módszer, réginek mondható, viszont pont ezért kisebb a teljesítmény igényük és képesek, egymás mellett párhuzamosan, működni, kombinált eredményük pedig azt mutatja, hogy együtt jól használhatóak.

Ábrajegyzék

1. ábra: [Yang02]
2. ábra: [Yang02]
3. ábra: http://www.cctvnews.co.kr/atl/view.asp?a_id=591
4. ábra: http://www.cctvnews.co.kr/atl/view.asp?a_id=591
5. ábra: [Kövér05]
6. ábra: [Bors03]
7. ábra: [Gao02]
8. ábra: [Kövér05]
9. ábra: [Inseong04]
10. ábra: <http://hackspc.files.wordpress.com/2009/07/bananascreen1.png>
12. ábra: <http://www.codeproject.com/Articles/27125/Ultra-Rapid-Object-Detection-in-Computer-Vision-Ap>
13. ábra: [Viola01]

Irodalomjegyzék

[Banana12]

A www.keylemon.com egy korábbi terméke (Utolsó látogatás: 2012 április)

[Bors03]

Bors, B., Horváth, R., Safranka, M., Vámosy, Z., „CERBERUS Project, Human Verification: a Biometric Approach”, in. Proc. IEEE International Conference on Computational Cybernetics, Siófok, Aug. 29-31, 2003

[Brunelli93]

R. Brunelli, T. Poggio, “Face Recognition through Geometrical Features”, Istituto per la Ricerca Scientifica e Tecnologica, Trento, Italy, 1993

[Brunelli93b]

R. Brunelli, T. Poggio, „Features versus templates”, IEEE Trans. Pattern Analysis and Machine Intelligence vol. 15, no. 10, pp. 1042 – 1052, October 1993

[Cox96]

Cox, I. J., Ghosn, J., Yianilos, P. N. 1996. Feature-based face recognition using mixture distance. In Proceedings, IEEE Conference on Computer Vision and Pattern Recognition. 209–216.

[DeMers93]

D. DeMers and G. Cottrell, S. J. Hanson, J. D. Cowan, and C. L. Giles, "Non-linear dimensionality reduction", Advances in Neural Information Processing Systems 5, pp.580 - 587 1992

[Gálfi08]

Gálfi Csongor, „Arcfelismerés számítógéppel”, Szabadkai Műszaki Szakfőiskola, 2008

[Gao02]

Y.Gao, M.K.H. Leung, „Face Recognition Using Line Edge Map”, IEEE Trans. Pattern Analysis and Machine Intelligence vol. 24, no. 6, June 2002

[Jesorsky01]

Oliver Jesorsky, Klaus J. Kirchberg, Robert W. Frischholz, Robust Face Detection Using the Hausdorff Distance, Berlin, June 2001

[Kanade73]

Kanade, T. 1973. Computer recognition of human faces. Birkhauser, Basel, Switzerland, and Stuttgart, Germany.

[Kim04]

Inseong Kim, Joon Hyung Shim, and Jinkyu Yang, „Face Detection”, Stanford University, 2004

[Kövér05]

Kövér, T., D. Vigh, and Z. Vámosy, "Improved Face Recognition in the MYRA System", 4th Serbian-Hungarian Joint Symposium on Intelligent Systems, Subotica, Serbia, pp. 187-195, September, 2006

[Madai12]

Madai Péter, Szigetvári Áron- Bevezető az arcfelismerés témájához

<http://www2.mit.bme.hu/services/vimm3241/tanul/beadott/regi/SzigetvariMadai/main.htm> (utolsó látogatás: 2012 április)

[Park05]

B.-G. Park, K.-M. Lee, and S.-U. Lee, „Face Recognition Using Face-ARG Matching”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.27, no.12, December 2005

[Reco12]

Recoware Kft. – Biometrikus azonosításról

http://www.recoware.hu/biometria/biometriai_azonositas/biometriai_azonositas_kezdolap.html (utolsó látogatás: 2012 május)

[Rowley98]

Rowley, Baluja, and Kanade, „Neural Network-Based Face Detection” , IEEE Trans. Pattern Analysis and Machine Intelligence, vol.20, no.1, January 1998

[Spacek08]

<http://cswww.essex.ac.uk/mv/allfaces/index.html>

(Utolsó látogatás: 2012.12.22)

[Turk91]

Matthew Turk, Alex Pentland “Eigenfaces for Recognition” Journal of Cognitive Neuroscience. Vol3. No. 1. 71-86, 1991

[Viola01]

Paul Viola, Michael Jones, „Rapid Object Detection Using a Boosted Cascade of Simple Features” Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference, vol.1, 2001

[Yang02]

M.-H. Yang, D.J.Kriegman, and N. Ahuja, „Detecting Faces in Images: A Survey”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34-54, January 2002