Óbudai Egyetem
Neumann János Informatikai Kar

Számítógépes képfeldolgozás

Arcdetektálás és kitakarás

Hallgató neve: **Héthy Zoltán**
Azonosítója: **JW3L34**
Szak megnevezése: **Mérnök Informatikus MSc**
Aktuális tanév: **2016/2017**

Objektum és arcdetektálás

Az objektumdetektálás feladata, hogy egy képről megállapítsa, hogy tartalmaz-e egy adott objektumot vagy sem, és ha tartalmaz, akkor megadja az objektum helyét és kiterjedését. Számos módszer létezik objektumdetektálásra, mint például a megjelenés-alapú, tudás-alapú vagy mintaillesztő eljárások. A leghatékonyabb általában a megjelenés-alapú szokott lenni, amely során úgynevezett pozitív és negatív képhalmazt használunk a tanítás során. Arcdetektálás esetén a detektálandó objektum egy arc lesz, ilyenkor a tanítás során a pozitív halmaz arcképeket, a negatív pedig olyan képeket tartalmaz, amelyeken az arcon kívül bármi szerepelhet. [1]

Az arcdetektálás eléggé használatos és elterjedt dolognak számít a mai világban. Ez egy olyan technológia, amellyel egy digitális képen automatikusan találhatunk meg arcokat. Az arcdetektálási vizsgálatok alapját gyakran a következő jellemzők képezik, mint például a fejforma, az arc részleteinek geometriája és a bőrszín. Rengeteg digitális berendezésben és rendszerben megtalálható már ez a technika. Léteznek olyan digitális fényképezőgépek, kamerák, amelyek beépített arcdetektálással rendelkeznek (1. ábra), de a legtöbb okostelefonban is megtalálható már ez a funkció. Továbbá ami, ennél még fontosabb, az arcfelismerő rendszereknek is az alapjául szolgál. Ilyen arcfelismerő rendszereket többnyire épületek, termek beléptető rendszereiben használnak, ahol fontos a biztonság, mint például repülőterek, bankok, stb.. Továbbá számítógépek belépésénél is előszeretettel alkalmazzák egyesek, de talán a legfontosabb felhasználási területe a számítógépes arcadatbázis, amely segítségével a rendőrségi nyomozások során az elmentett arcok összehasonlíthatók és azonosíthatók. [2] [3]

1. ábra: Arcdetektálás fényképezőgéppel

Arcdetektálás problémái

Az ember arcának detektálása számítógép segítségével nem oly egyszerű feladat, mint ahogyan az az emberi arcfelismerés során történik, mivel a számítógép csak egy szenzorokból áradó pixelhalmazt lát, R G B komponensek dzsungelét, amiben rengeteg nehezítő tényező szerepelhet:

- **A kép variációi** - a képek sokféle képpen variálódhatnak, a rajtuk lévő arcoktól függetlenül. (pl. a kép méretaránya, felbontása, minősége, kontrasztja, stb.)
- **A póz** - ide tartozik az arcok iránya és az arcok kamerától való távolsága, továbbá ha egy képen több arc is látható. Ezek a kétdimenziós képeken különböző perspektivikus torzítással jelenhetnek meg, amely többféle arc-méretet és arányt eredményezhetnek.
- **Megvilágítás és textúra** - a fényforrások mozgása és fényerejük megváltozása hatalmas változásokat képes előidézni egy arc megjelenésén, aminek köszönhetően különböző árnyékolásokat kaphat. A különféle textúrákhoz tartoznak például az arcszőrzet vagy arcok közötti eltéréseket. (bőrhibák, stb.)
- **Háttér** - a háttér szokta a legnagyobb kihívást jelenteni, hogy a számunkra lényeges objektumot megfelelő biztonsággal elválaszthassuk a hátterüktől a későbbi arcfelismerés megvalósításának elsődleges feladataként.
- **Alakvariációk** – ide tartoznak az arckifejezések (nyitott vagy csukott száj), illetve az illető fej- és arcformája. [3]

Arcdetektáló technikák

Az arcdetektáló módszereket, működésük alapján négy csoportba sorolhassuk: 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 arcmintákat kapunk ugyan arról a képről. Tehát az algoritmus először az eredeti kép egy kis felbontású változatán arcmintákat, majd növelve a felbontást az adott szinthez tartozó arcmintákat 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.

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.

Mintaillesztés alapú technikák

Ezen technikák azon alapulnak, hogy egy előre elkészített arcmintát keresünk a képen. Ez lehet egy élekkel 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 elő feldolgozásnak 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

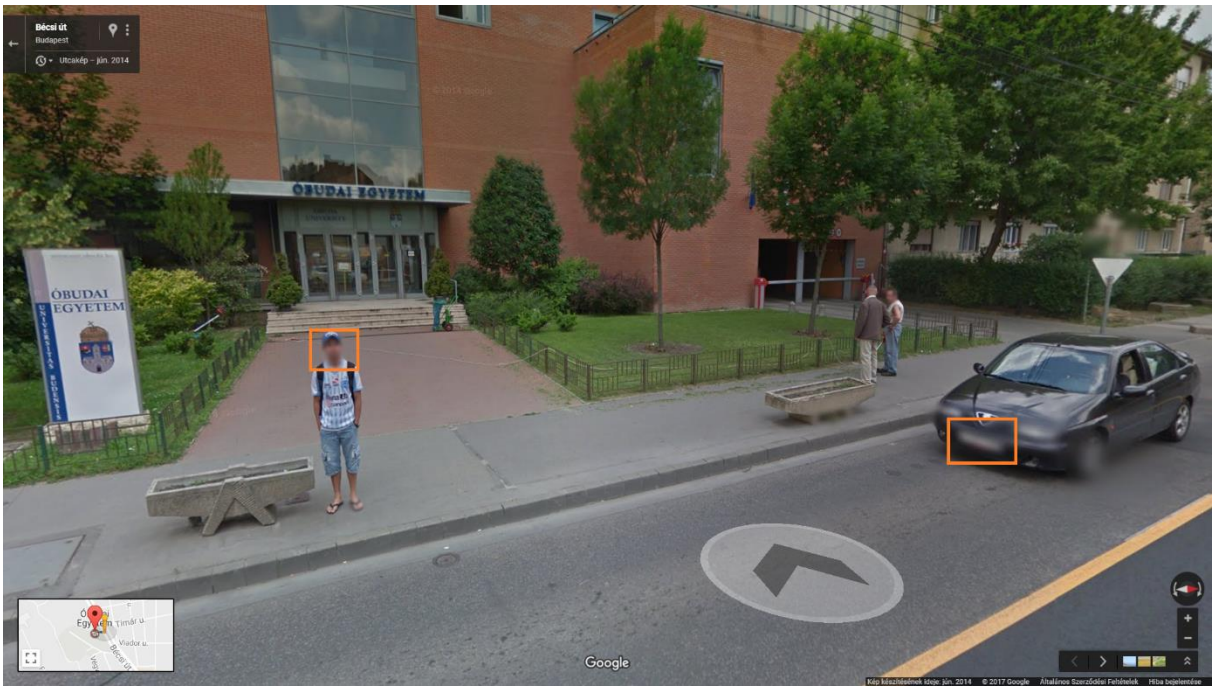
Egy előre elkészített tanító mintákat tartalmazó halmaz alapján készítenek egy „modellt", és ennek a modellnek megfelelő részeket keresnek a képen. Az ilyen módszereknél jellemzően neurális hálózatot is használnak.
Rowley, Baluja, és Kanade 1998-ban előálltak egy neurális hálózat alapú arcdetektáló rendszerrel. 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.
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. 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. A szín alapú megközelítés további hátránya, hogy megvilágításra érzékeny, és önmagában nem elegendő a detektáláshoz. Előnye viszont, hogy forgatásra, eltolásra, átskálázásra invariáns. [4]

Feladat ismertetése

A számítógépes képfeldolgozás tárgy féléves feladataként egy olyan arcdetektáló szoftvert készítettem el, amely előbb felismeri a képen található arcokat, majd azokat különböző módszerek segítségével cenzúrázza a képeken. A detektált arcok kitakarására három különböző módszert használtam:

- arc elmosása (Gauss-szűrő segítségével)
- arc negálása
- arc lecserélése egy előre megadott képre

Ezek az eljárások tökéletesen elérik a kívánt hatást, miszerint a cél az arcok minél jobban felismerhetetlenné alakítása. Kép részleteinek cenzúrázása elég gyakori dolog a médiában, amit álltalában a nemzetbiztonság, a közérdek vagy a „jó ízlés” ürügyében alkalmaznak. Azonban egy konkrétabb példának megemlíthető a Google Maps, ahol a Google a személyi jogok fenntartása érdekében az utca képeken elhomályosításokat használ az emberi arcok és az autó rendszámok helyén (2. ábra).



2. ábra: Google Maps utca kép

Az arckitakaró szoftvert C# nyelven lett megvalósítva, az arcdetektálás részéhez pedig az Emgu CV könyvtárcsomagot használtam, ami nagyban megkönnyítette ennek a megvalósítását.

OPEN CV és EMGU CV

Az OPEN CV az Intel által fejlesztett olyan programozási funkciókat tartalmazó könyvtár, amely segítségével a képfeldolgozó eljárások hatékonyan valósíthatók meg. A könyvtárcsomag nyílt forráskódú, szabadon felhasználható. A könyvtárcsomagot eredetileg C nyelven írták, majd a 2.0-ás verzióban megjelent a C++ támogatás, ami később szinte ki is váltotta a C-t. Jelen pillanatban az OPEN CV legújabb elérhető verziója a 3.2-es, amely még több és még gyorsabb képfeldolgozó eljárásokat tartalmaz. Természetesen nem csak ezen a nyelven lehet fejleszteni a képfeldolgozás és gépi látás témakörben, ugyanis az OPEN CV-hez fejlesztettek úgynevezett „wrapper”-eket, amelyek segítségével már C#-ban, Python, Ruby, és Java nyelveken is használható a csomag. Továbbá léteznek hasonló könyvtárcsomagok, mint például az Aforge.Net, VXL, vagy az Integrated Vision Toolkit (IVT), amelyekkel hasonló problémákat oldhatunk meg.
Az EMGU CV tulajdonképpen egy Open CV C# wrapper, ami annyit takar, hogy Open CV-s műveleteket C# nyelven is írhatunk. Ehhez ugyanúgy tartoznak könyvtárak, pusztán annyi különbséggel, hogy az eredeti Open CV dll fileokat is mellékelni kell a projekthez. Az EMGU CV a következő rendszereken futtatható: Windows, Android, Maemo, FreeBSD, OpenBSD, iOS, Linux és Mac OS. [5]

EMGU CV telepítése és konfigurálása

Ahhoz hogy az EMGU CV kódokat használni tudjam a DotNet projektemben, a következő lépéseket kellett végrehajtanom:

- Az operációs rendszernek megfelelő EMGU CV csomag letöltése az alábbi linkről: https://sourceforge.net/projects/emgucv/files/emgucv/
  (Nem a legújabb verziót, hanem a 2.4.2-t használom)
- A könyvtárcsomag telepítése (kicsomagolása) adott helyre (C:\Emgu mappa)
- A szükséges dll-ek beimportálása a DotNet projektünkbe az Emgu CV bin mappájából (Ezeket át is kell másolni a saját projektünk bin/Debug mappájába, hogy más eszközön való futtatás esetén is elérhetőek legyenek)
- Az EMGU CV modulok hozzáadása a projektben használt referenciákhoz (Add Reference…)
- Végül az EMGU CV névterek megadása a programunkban a using résznél

Az alkalmazás megvalósítása

Az arc kitakarásra egy ablakos alkalmazást hoztam létre, aminek a felhasználó felülete (3. ábra) egyszerűen és könnyen kezelhetően lett kialakítva. Lehetőség van rajta kép beolvasására, arcdetektálásra a beolvasott képen és a detektált arcok kitakarására. A kitakarás típusát egy legördülő menü segítségével választhatjuk ki. Egy pictureBox-on jelenítjük meg a beolvasott és már manipulált képeket is. Az egyes funkciók futási idejét pedig az ablakos alkalmazás alján található állapotjelzőben jelenítjük meg.

3. ábra: Felhasználói felület

Főbb funkciók megvalósítása

A kép beolvasása egyszerűen az *OpenFileDialog* osztály segítségével történik, amelyben csak bmp, jpg és png típusú képek beolvasása engedélyezett. Bármilyen méretű képet be tudunk olvasni, mivel az alkalmazás a pictureBox méretéhez nyújtja a kép méretét.

Az arcdetektálás az EMGU CV könyvtár segítségével történik, amelyben implementálva van Viola-Jones féle objektum detektor. Ez az algoritmus Haar maszkokat használ az arcok megtalálásához. Ezeket a Haar maszkokat xml fájlokból olvassa be a programunk. Mivel az arcdetektálás mellett próbálkoztam még szemdetektálással is, ezért két darab xml fájlunk lesz. A *"haarcascade_frontalface_default.xml"* fájlban az arcjellemzők, a *"haarcascade_eye.xml"* fájl pedig a szemjellemzőket fogja tartalmazni.

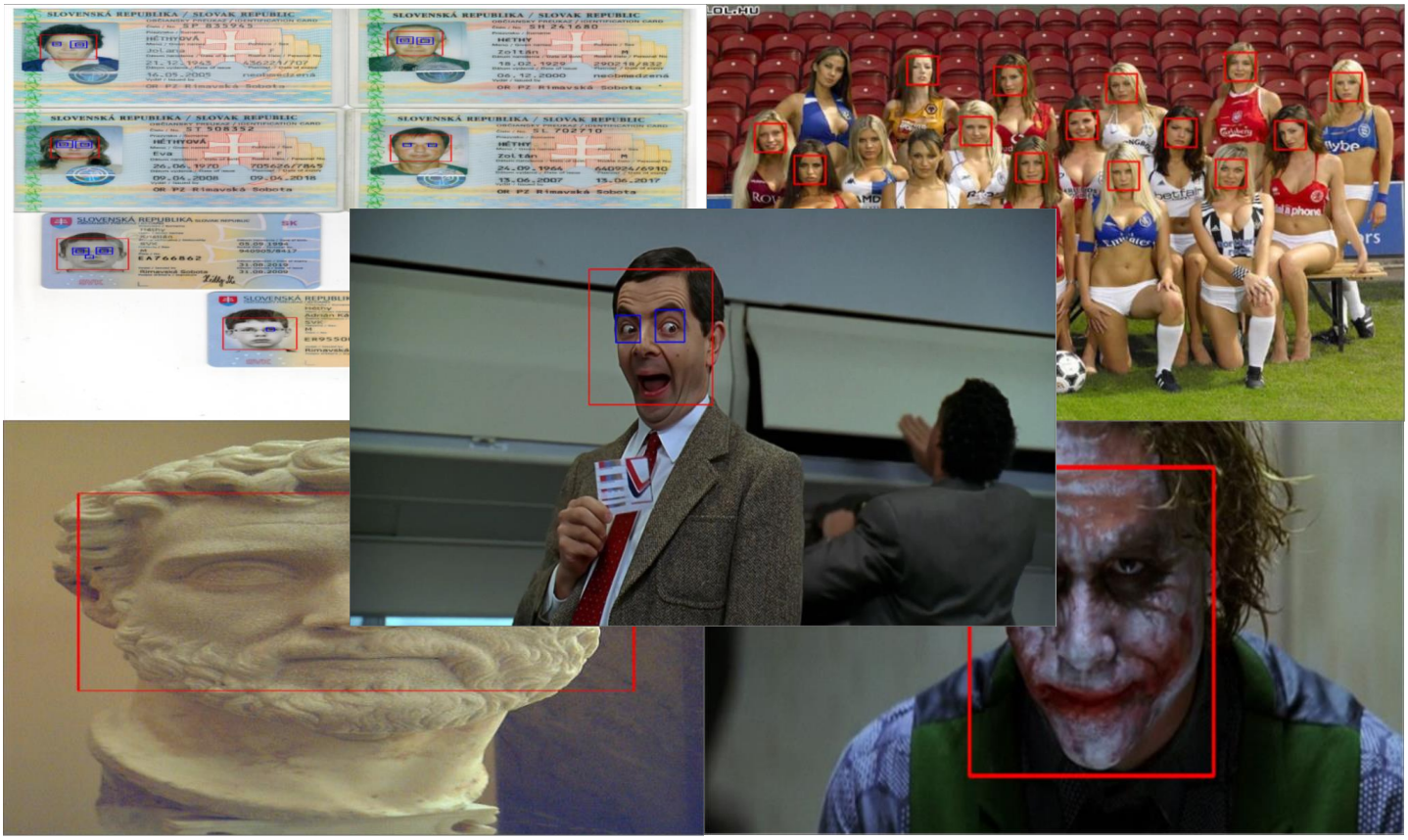
Az arcdetektálás előtt szükség van valamiféle előfeldolgozásra is, amivel a lényegtelen információkat, zajokat tüntetjük el, a lényegeseket pedig kiemeljük, ezzel is növelve algoritmus hatékonyságát. Első lépésben szürkeárnyalatossá alakítjuk a képet, majd egy hisztogram kiegyenlítést hajtunk rajta végre, amivel normalizáljuk a fényességet és növeljük a kontrasztosságot. Arcdetektálás során a detektált arcokat téglalapként tároljuk egy listában, amikből aztán a későbbiekben kinyerhetjük a szükséges paramétereket, mint például a pozícióját és a méreteit. Ezek alapján tudjuk megrajzolni a téglalapokat az arcok köré és az arckitakarásnál is szükségesek.

Az arcok kitakarása három féle képpen történhet. Az elmosáshoz Gauss szűrőt, a negáláshoz Invert szűrőt, az arccseréhez pedig egyszerűen egy képet adunk meg. Az első kettőnél először kivágjuk a detektált arcot a képből, amin aztán végrehajtjuk a megfelelő transzformációt, utána pedig visszarajzoljuk az eredeti helyére a képen. Az arccsere estében nincs szükség kivágásra, ott csak egyszerűen kirajzoljuk az adott képet a detektált arc helyén a megfelelő méretben.

Tesztelés

A program tesztelését különböző képeken végeztem, amelyek általában valamilyen arcot tartalmaztak. A rend kedvéért teszteltem pár olyan képen is, ami nem tartalmazott arcot. Ezeknél sohasem detektáltunk arcot, tehát ebből a szempontból az az algoritmus tökéletesen működött, nem adott false eredményeket. Azonban voltak olyan képek, amelyek tartalmaztak arcot, viszont az algoritmus nem ismerte fel azokat. Ezen problémák kiváltó okai többnyire a kép alacsony felbontása, rossz minősége, arc profilból vett nézete, nem emberi arc tesztelése voltak.
Az alkalmazás több arc detektálására is képes egy képen, azonban nem garantált az összes arc felismerése. Nagyobb felbontású képeknél hatékonyabban működött, viszont ezzel együtt növekedett a futási idő is. Érdekesség képpen akadtak olyan nem valós arcok is, amelyeket felismert az algoritmus, mint például egy élethű anime karakter arca vagy egy szobor feje.
Az egészet összegezve a legfontosabb feltétele az arcdetektálásnak az volt, hogy a képen élethű, szemből ábrázolt arcok szerepeljenek.

Képek amiken talált arcot



Képek amiken nem talált arcot

Felhasznált irodalom

[1] Szilágyi, A. *Interaktív információs képfeldolgozó panel algoritmusai.* 2010. [pdf].

[2] Tóth, S. *Innovatív digitális kameramegoldások*. 2015. [online]. Elérhető az interneten: https://www.magyar-elektronika.hu/34-tartalom/tartalom/1287-innovativ-digitalis-kameramegoldasok-

[3] Kövér, T. – Vígh, D. *Arcdetektáló- és felismerő Rendszer.* 2005. [pdf].

[4] Varga, M. *WATCHR – Azonosítás arcfelismeréssel.* [pdf].

[5] Jako, D. *Computer Vision*. [online]. Elérhető az interneten: http://vip.tilb.sze.hu/~wersenyi/JakoD.pdf