OKOSVÁROS-TECHNOLÓGIÁK IV.

Számítógépes látás a közigazgatásban

Alapfeladatok és alkalmazások

SZEMENYEI MÁRTON

Dialóg Campus

Számítógépes látás a közigazgatásban
Alapfeladatok és alkalmazások

Okosváros-technológiák
A technológia fejlődésének irányai és hatása
IV. kötet

Sorozatszerkesztő
Sallai Gyula

Szemenyei Márton

SZÁMÍTÓGÉPES LÁTÁS A KÖZIGAZGATÁSBAN

Alapfeladatok és alkalmazások

Dialóg Campus Kiadó ❖ Budapest, 2019

A mű a KÖFOP-2.1.2-VEKOP-15-2016-00001 azonosítószámú, „A jó kormányzást megalapozó közszolgálat-fejlesztés” elnevezésű kiemelt projekt keretében működtetett 2017/162/BME-VIK számú, Okos város – okos közigazgatás elnevezésű Államtudományi Kutatóműhely keretében, a Nemzeti Közszolgálati Egyetem felkérésére készült.

Lektor:
Jakab László

Tartalom

Vákát oldal

Bevezetés

A különféle számítógépes látórendszerek az élet egyre több területén váltak elterjedtté. Ez a rendelkezésre álló számítási teljesítmény és az eszközök rohamos gyarapodásának, valamint a rendszerekben használt algoritmusok jelentős fejlődésének is köszönhető. Ezen algoritmusoknak számos felhasználási területe létezik a robotikától és az automatizálástól kezdve a virtuálisvalóság- és kiterjesztettvalóság-rendszereken keresztül egészen a szórakoztatóiparig. Jelentős továbbá e módszerek közszolgálati területeken történő alkalmazásának lehetősége is.

A számítógépes látás kutatási területének az a célja, hogy algoritmikus eszközök segítségével képekből vagy képsorozatokból egy másik döntéshozó alkalmazás vagy személy számára releváns információk legyenek kinyerhetők. Az elvégzendő feladat legnagyobb nehézsége, hogy akár egyetlen kép is milliós nagyságrendű adatból (képpontból) áll, amelyek ráadásul ennél is számos nagyságrenddel több konfigurációban alkothatnak képeket. Ilyen adatmennyiség feldolgozásához hatékony algoritmusokra és nagy teljesítményű eszközökre van szükség.

A terület további nehézsége, hogy habár az ember képes egy látott kép alapján számos létfontosságú információt meghatározni (sőt, a szem az ember legfontosabb érzékszerve), e feldolgozás nagy része tudat alatt zajlik, így nem lehet ezeket a képességeket könnyen egzakt algoritmusra váltani. Ezt a problémát tovább nehezíti, hogy feltehetően számos látásalapú döntésünkhöz felhasználunk olyan információkat is, amelyekhez egy másik érzékszervünk segítségével jutottunk. A fenti problémák miatt a számítógépes látás területén gyakorta alkalmazunk heurisztikákra, illetve gépi tanulásra, matematikai optimalizálásra épülő eljárásokat, amelyeknek a minden eshetőségre való helyes működését nem tudjuk garantálni.

Ezeket az algoritmusokat számos különböző módon csoportosíthatjuk. Ezek közül az egyik legelterjedtebb az eljárások célja (kimenete) alapján történő csoportosítás. Itt megkülönböztetjük a képfeldolgozás (angolul: image processing), valamint a számítógépes látás (angolul: computer vision) algoritmusait. A képfeldolgozás esetén a célunk az, hogy az algoritmus eredményeként egy olyan új képet kapjunk, amely valamilyen szempontból

számunkra előnyösebb, mint az eredeti kép volt. Ezek az eljárások gyakorta hivatottak a képek további feldolgozását vagy éppen az ember általi láthatóságát segíteni.

A számítógépes látás algoritmusai ezzel szemben kimenetükön nem egy újabb képet, hanem valamilyen, a bemenetből kinyert, magasabb absztrakciós szinten létező információt szolgáltatnak. Ezenfelül külön meg szoktuk különböztetni azt az esetet, amikor ezeket az eljárásokat egy beágyazott rendszerben (gép, autó, robot, telefon stb.) használjuk, amely esetben gépi látásról (angolul: machine vision) beszélünk. Elterjedt kifejezés továbbá a videoanalitika, amely esetben az algoritmus bemenete egy képsorozat.

A számítógépes látáson belül gyakorta meg szoktuk különböztetni azokat a megoldásokat, amelyek a gépi tanulás (angolul: machine learning) algoritmusait használják a működésük során; ezt a területet tanuló látásként (angolul: learning vision) is definiáljuk. Ezeken belül külön figyelmet érdemelnek azok az eredmények, amelyek az elmúlt néhány évben rendkívül népszerűvé vált mélytanulás (angolul: deep learning) megoldásait alkalmazzák. A terület többi módszerét – sokszínűségük ellenére – hagyományos látásnak nevezzük.

A számítógépes látás alkalmazása területén egy rendkívül jelentős kérdés a valósidejű működés megoldása. A *valósidejűség* fogalma azt jelenti, hogy egy adott algoritmus esetén garantálni tudjuk, hogy az egy adott időn belül mindenképpen befejezi a működését. A számítógépes látás esetén azonban nemcsak a gyors működést és az abból következő gyors reakcióidőt szeretnénk garantálni, hanem azt is, hogy az adott algoritmus az időkorlát miatt a megbízhatóságából ne veszítsen. Továbbá, mivel a számítógépes látás módszerei meglehetősen nagy számításigényűek, ezért a megfelelően gyors működéshez gyakran nagyságrendbeli gyorsításra van szükség.

Ezt a mértékű sebességnövelést leggyakrabban speciális feldolgozó eszközök segítségével érjük el. Amennyiben a feldolgozást lokálisan, egy mobilis eszközön végezzük el (például intelligens kamerák, mobil készülék), akkor gyakran használunk digitális jelfeldolgozó egységet, azaz DSP-t (digital signal processor). Ezek tipikusan relatíve alacsony fogyasztású eszközök, amelyeknek a belső architektúráját oly módon alakították ki, hogy nagy mennyiségű adat feldolgozására legyenek alkalmasak. Hasonló esetekben működtetnek még alkalmazásspecifikus integrált áramköröket vagy programozható hardveregységeket is.

Abban az esetben, ha a feldolgozás nem lokálisan, hanem egy távoli gépen történik, érdemes grafikus feldolgozó egységeket, azaz GPU-kat (graphical processing unit) alkalmazni. Bár ezeket az eszközöket eredetileg grafikus alkalmazások (elsősorban videojátékok) támogatására fejlesztették ki, a hardverarchitektúrájuk kiválóan megfelel számos látó algoritmus futtatására is. Manapság a GPU alapvető eszköz például a mélytanulás, valamint a kriptovaluták terén. Mivel ezen eszközök fogyasztása jelentős, ezért leginkább csak asztali, illetve szervergépekben találhatók meg, ezért az árért viszont akár több százszoros gyorsítás is elérhető velük.

A számítógépes látás közigazgatásban való alkalmazásáról szóló írás jelenlegi, első első része betekintést ad a képfeldolgozás és a számítógépes látás alapfeladataiba és az azok megoldására használatos módszerekbe. Ezenfelül az egyes alapfeladatokhoz egyszerű alkalmazási példákat mutatunk be.* Az írás második kötetében részletezzük a modern feladatokat, például a háromdimenziós látást, valamint a mélytanulás-alapú látást, továbbá a kötet ismertetni fog komplex alkalmazásokat is.

* A kötet szerzője: Szemenyei Márton, a Budapesti Műszaki és Gazdaságtudományi Egyetem tanársegéde, kutatásai a mesterségesintelligencia-alapú számítógépes látásra fókuszálnak. Munkája során részt vett a RoboCup nevű évente megrendezett nemzetközi robotikai versenyen.

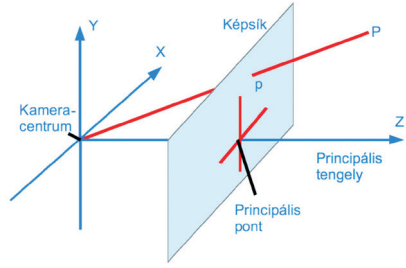
Vákát oldal

1. Képek készítése

A számítógépes látás legalapvetőbb feladata a feldolgozandó képek készítése. Bár a kamerák közismert, hétköznapi eszközök, ezek működésének és fontosabb tulajdonságainak ismerete elengedhetetlen ahhoz, hogy az adott alkalmazáshoz megfelelő eszközt válasszunk. Ebben a fejezetben a különböző szenzortípusokat és az azok közti különbségeket mutatjuk be.

A kamerák működési elve nagy részben az emberi szem működési elvén alapszik. Az emberi szem egy üreges gömbtest, amelynek az elülső részén egy nyílás, az írisz található, amelyen keresztül fény áramlik be a közvetlenül mögötte elhelyezkedő lencsébe. A lencse a párhuzamosan beérkező fénysugarakat egy pontba fókuszálja, amely a szem hátulján lévő fényérzékelő hártyán, a retinán helyezkedik el. A lencsének köszönhetően az egy irányból érkező fénysugarak a retinán pont ugyanarra a helyre érkeznek, így az emberi látás éles lesz.

A retina felületén számos fényérzékelő sejt helyezkedik el, amelyek továbbítják az ingereket az idegrendszernek. Fényérzékelő sejtből kétféle létezik: a gyakoribb pálcikák csupán a fényintenzitást érzékelik, mivel a látható fény teljes tartományán hasonló érzékenységűek. Ezzel szemben a ritkábban elhelyezkedő csapok a kék, a zöld vagy a piros szín frekvenciatartományában érzékenyek csak, lehetővé téve a színek érzékelését.



1. ábra

A pinhole kameramodell matematikai ábrázolása

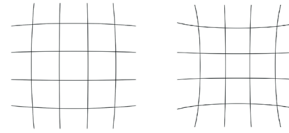
Forrás: a szerző szerkesztése

A számítógépes látásban gyakran használt a pinhole kameramodell,[1] amely esetében a fény egy apró lyukon (pinhole) keresztül jut be egy üreges dobozba, amelynek a hátsó falára verődve a fénysugarak egy fordított állású képet alkotnak. A kamerákban a fényszenzorok kétdimenziós mátrixelrendezésben ezen a hátsó falon találhatók. A pinhole és a szenzortömb közötti távolságot nevezzük a kamera fókusztávolságának (f), míg a szenzortömbre merőleges, a lyukon átmenő tengelyt pedig a kamera principális tengelyének. A szenzortömb és a principális tengely metszéspontja a principális pont (px, py). Ezen mennyiségek segítségével a kamera a háromdimenziós objektumokat az alábbi képlet alapján képezi le a képsíkra.

$$x = -f\frac{X}{Z} + p_x \quad y = -f\frac{Y}{Z} + p_y$$

A valóságban a kamerákban egy valamelyest módosított elrendezést használunk, ugyanis a pinhole túlságosan kicsi ahhoz, hogy azon a jó minőségű képalkotáshoz elegendő fény áramoljon be. Ha a pinhole méretét növelnénk, akkor pedig a fénysugarak nagyobb szóródása miatt homályos képet kapnánk. Éppen ezért a kamera bemeneti nyílásához egy lencsét teszünk, amely egy pontba fókuszálja a párhuzamosan beérkező fénysugarakat, így helyettesíti a pinhole-t, mérete azonban már elég nagy ahhoz, hogy megfelelő mennyiségű fényt engedjen át.

A lencse hozzáadása azonban komplikációt okoz: a kamera látóterének a szélsőbb részeiből érkező képsugarak már nem párhuzamosan fognak beérkezni a lencsére, így másképp fognak megtörni, mint a párhuzamosan érkező sugarak. Ennek következményképp ezek a fénysugarak a fényérzékelő szenzortömbre más pozícióban fognak becsapódni, így a képen is arrébb kerülnek. Ezt a jelenséget radiális torzításnak hívjuk, és gyakori jelenség, különösképpen széles látószöggel rendelkező kamerák esetén.[2]



2. ábra
A radiális torzítás hatása
Forrás: a szerző szerkesztése

[1] Kató Zoltán – Czúni László (2001): *Számítógépes látás.* Budapest, Typotex.
[2] Russ, John C. (2011): *The Image Processing Handbook.* Boca Raton, CRC Press.

Fontos megjegyezni, hogy a leképzés során az objektum méreteire vonatkozó háromdimenziós információ egy része elveszik, és csak komplex rekonstrukciós technikákkal becsülhető meg. Ahhoz, hogy ezeket az információkat megbecsülhessük, ismernünk kell a kamera paramétereit (fókusztávolság, principális pont) és a torzítását. Sok esetben ezek a paraméterek a kamera adatlapján megtalálhatók, azonban számos esetben ezeket nekünk kell kamerakalibrációs eljárások segítségével meghatároznunk.[3]

A kamerakalibrációs eljárások során a kamera segítségével képeket készítünk egy előre ismert kalibrációs objektumról. Ez az objektum többféle lehet, azonban az esetek túlnyomó részében egy egyszerű, nyomtatottsakktábla-mintázatot használunk. A pontos, megbízható eredmények érdekében célszerű különböző szögből és távolságból több tucat képet készíteni. Ezt követően egy algoritmus detektálja a sakktábla sarkait a képen, majd a sakktábla ismert méreteinek és a képén meghatározott pontoknak a segítségével a kamera paraméterei és torzításai statisztikai és optimalizálási módszerek segítségével meglehetősen pontosan megbecsülhetők.

A kamera fizikai felépítésén túl rendkívül fontos jellemző a szenzortömbben található fényérzékelő eszközök típusa. Fényszenzorok terén a két elterjedt megoldás a CCD-érzékelő (charge-coupled device), valamint az aktív CMOS-érzékelő (complementary metal-oxide semiconductor). A CCD-szenzor egyes elemei analóg eszközök, amelyek fotonok becsapódása esetén elektromos töltést tárolnak el. A kép készítéséhez szükséges idő eltelte után az egyes sorok legszélső cellája átadja a töltését a kimeneti erősítőnek, majd a cellák átadják töltésüket a szomszédos cellának. A fent leírt működési elv alapján a CCD-szenzor kiolvasása soronként, a sorokon belül pedig elemenként történik.[4]

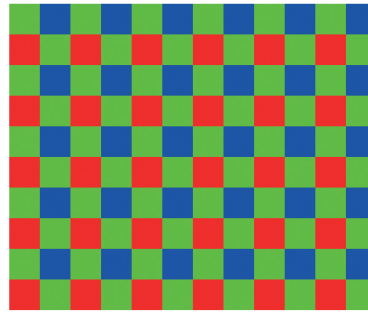
A CMOS-érzékelők esetén ezzel szemben minden cellára jut egy erősítő, így a szenzortömb kiolvasása lényegesen gyorsabb. Ennek viszont az az ára, hogy az erősítők helyet foglalnak a szenzortömbben, amely egyébként a fény elnyelésére szolgált volna. E hátrány kiküszöbölésére a CMOS-cellákra mikrolencséket tesznek, amelyek az érzékelőre terelik azokat a fotonokat, amelyek egyébként az erősítőre csapódtak volna be. Kisebb mérete, fogyasztása, valamint kedvezőbb ára miatt a kamerák többsége

[3] Kató–Czúni 2001.

[4] Russ 2011.

CMOS-érzékelőket használ. A CCD-szenzorok leginkább csúcsminőségű videokamerák terén elterjedtek.[5]

A kamerák színérzékelésének kialakítására is több megoldás létezik, amelyek közül a Bayer-szűrőn alapuló a legolcsóbb és leginkább elterjedt. A Bayer-szűrő egy olyan tömb, amelynek egyes elemei csupán bizonyos színeket eresztenek át. Ezt a szűrőt a szenzortömb elé helyezve elérjük, hogy az egyes CCD- vagy CMOS-érzékelők is csak ezekre a színekre adjanak jelet. A Bayer-szűrő a legtöbb esetben kétszer annyi zöld színt áteresztő elemet tartalmaz, mint kéket és pirosat, aminek az az oka, hogy az emberi szem érzékenysége is hasonló.



3. ábra
A Bayer-szűrő elrendezése

Forrás: a szerző szerkesztése

Egy alternatív megoldás a 3CCD-szenzor, ahol egy prizma segítségével a három színkomponenst külön-külön szenzortömbre tereljük, amivel élesebb színelválasztást tudunk elérni. Mivel három külön szenzortömböt használunk, a 3CCD-megoldás alacsony megvilágítás esetén lényegesen jobban teljesít a Bayer-szűrőnél, mindezt természetesen magasabb ár mellett.

Kiolvasás után a szenzorok által készített kép a számítógépbe beérkezve egy kétdimenziós számhalmaz lesz. Ennek egyes elemei az adott pozícióba beérkező fény intenzitását jelölik. Ezeket az elemeket képpontoknak vagy pixeleknek nevezzük. A számítógépben a pixeleket a legtöbb esetben egy 8 bites számmal jellemezzük, ahol 0 jelenti a teljesen sötétet, míg a maximális 255-ös érték pedig a teljesen világos képpontot. Színes képek esetén

[5] Russ 2011.

minden pozícióhoz három számérték tartozik, amelyeket RGB (red-green-blue) rövidítéssel jelölünk.

Az elmúlt néhány évben egyre inkább elterjedtek olyan speciális szenzorok, amelyek az egyes pixelek intenzitása mellett azoknak a szenzortól számított távolságát is képesek meghatározni, így minden egyes képponthoz egy negyedik számértéket is hozzárendelnek. Ezeket az eszközöket RGB-D-nek vagy mélységkameráknak nevezzük, ahol a D az angol depth, vagyis *mélység* szóból származik.

E kameráknak alapvetően két változata létezik: az elsőt sztereokamerának nevezzük, ahol két, egymástól fix távolságra lévő kamera van egy házba építve, és az egyes pixelek távolságát a két készített kép közötti megfeleltetésekből számolhatjuk ki. Ezeket az eszközöket általában a gyártás során kalibrálják, valamint a mélység számítása a kamerába épített feldolgozó hardveren megtörténik.

Ezzel szemben az infravörös technológián alapuló mélységkamerák három elemből állnak: egy közönséges RBG-érzékelőből, egy infravörös vetítőből és egy, az infravörös tartományban működő érzékelőből. A működésük elve az, hogy az ember számára láthatatlan infravörös tartományban egy előre meghatározott mintázatot vetítenek ki, amelyet az infravörös érzékelő visszaolvas, és a mintázat torzulásából következtet a látott kép térbeli struktúrájára. A kettő közül az infravörös alapú érzékelők elterjedtebbek, mivel jobb minőségű eredményeket adnak, és kevesebb feldolgozást igényelnek. Hátrányuk, hogy a környezetben található egyéb infravörös források megzavarhatják az eredményeket.

4. ábra
A Kinect One RGB-D-érzékelő
Forrás: https://en.wikipedia.org/wiki/Kinect (A letöltés dátuma: 2018. 06. 05.)

Létezik még ezen felül az úgynevezett LIDAR-érzékelő.[6] Ez a radarral megegyező módon a visszavert elektromágneses hullámok késleltetéséből és frekvenciájából következtet az objektumok távolságára, csak éppenséggel rádióhullámok helyett lézersugarak segítségével működik. A digitális feldolgozóegységek válaszideje hatalmas javulásának köszönhetően ma már rendkívül pontos – néhány cm-es nagyságrendű – távolságokat tudunk e technológia segítségével mérni.

[6] Cracknell, Arthur P. – Hayes, Ladson (2007): *Introduction to Remote Sensing.* London, Taylor and Francis.

2. Képjavítás

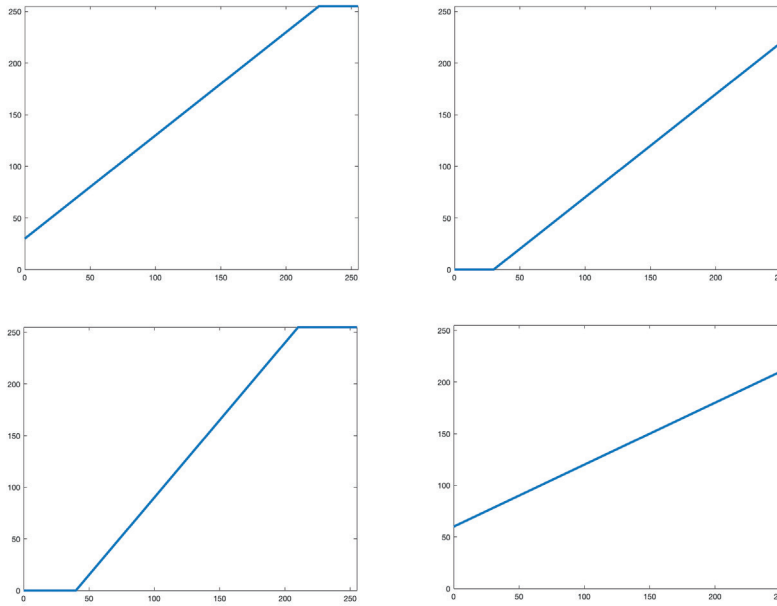
A különböző érzékelő eszközökből kinyert képek nem mindig teljesen megfelelők a felhasználási céljukra, aminek számos oka lehet. Gyakran a kép készítésekor a környezetet nem tudjuk kontrollálni, így a kép készítésének körülményeit nem lehet vagy legalábbis nehéz optimalizálni. Egy másik ok, hogy a szenzor, az adatátvitel vagy a tárolás hibáinak következményeképp a képen különféle zajok és hibák keletkeznek, amelyeket a feldolgozás elején érdemes kiküszöbölni. A fenti okokból kifolyólag a képjavítás a képfeldolgozás egyik legfontosabb alapfeladata. Ennek célja, hogy a képet akár a jobb láthatóság vagy a hatékonyabb további feldolgozás érdekében módosítsuk.

A képjavítás egyik legegyszerűbb módja a különféle intenzitástranszformációk[7] használata. E módszer alkalmazása során az algoritmus pixelenként végighalad a képen, és minden egyes képpont értékét egy előre meghatározott transzformációs függvény alapján megváltoztatja. Ez a transzformációs függvény az esetek túlnyomó többségében az új intenzitásértéket csupán az adott képpont korábbi intenzitása alapján határozza meg.

E transzformációknak több verziója létezik: amennyiben az egyes pixelek intenzitásához egy konstanst adunk hozzá, akkor a kép világosságát tudjuk változtatni (növelni vagy csökkenteni a konstans előjelétől függően). A pixelek világosságértékét egy konstanssal szorozni is lehetséges, ekkor a kép kontrasztosságát tudjuk csökkenteni vagy növelni. Kontrasztosságtranszformáció esetén egy konstans hozzáadása is megtörténik, hogy a kép új intenzitásértékeit középre toljuk. Fontos megjegyezni, hogy ezek a transzformációk feltételezik, hogy a pixelek értéke szaturál, azaz telítésbe megy, ha egy transzformáció azokat a minimális 0 érték alá vagy a maximális 255 érték felé vinné.

---

7 Russ 2011.

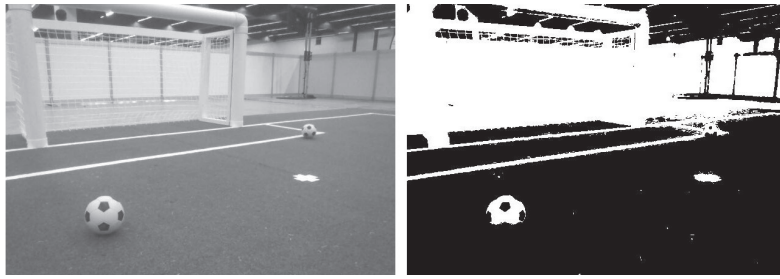


5. ábra

Különböző intenzitástranszformációk

Forrás: a szerző szerkesztése

Az intenzitástranszformációk egy különös fajtája a küszöbözés. Ekkor egy előre meghatározott küszöbérték egyik oldalán lévő pixeleket 0-ba, míg a másik oldalon lévőket 1-be állítjuk, így egy bináris képet kapunk. Ezt a műveletet el lehet végezni két küszöbérték segítségével is, ekkor a tartományon belüli vagy kívüli értékeket állítjuk 1-re. A küszöbözés egyik fő haszna, hogy a képről bizonyos intenzitású képpontokat ki lehet emelni, és azok mennyiségéből, méretéből vagy pozíciójából további következtetéseket vonhatunk le. Fontos megjegyezni, hogy egy előre meghatározott küszöbérték használata esetén a fényviszonyok megváltozása a módszer eredményét is nagymértékben befolyásolhatja, így a gyakorlatban gyakran használunk adaptív – az adott képen lévő intenzitások eloszlásából meghatározott – küszöbértéket.



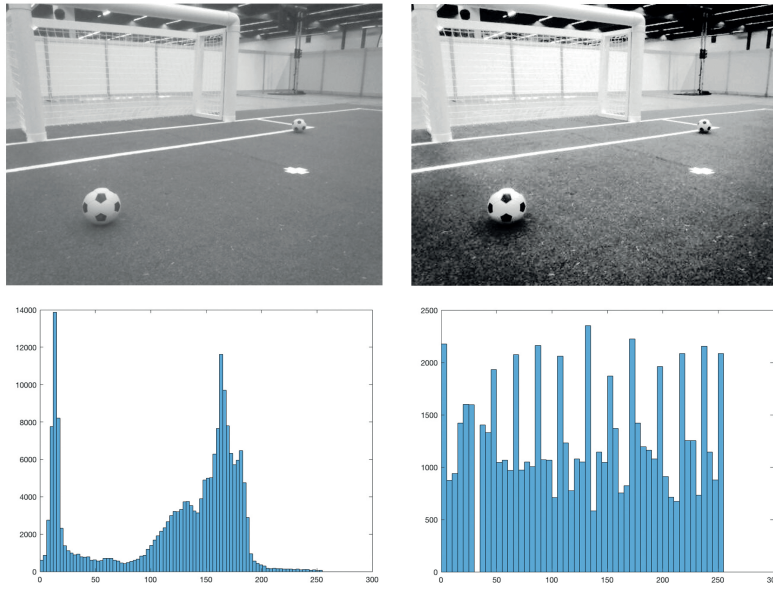
6. ábra
Az eredeti és a küszöbözött változata
Forrás: a szerző felvétele és szerkesztése

Intenzitástranszformációkat többcsatornás (vagyis színes) képek esetén is használhatunk. Ekkor az egyes színcsatornákra egymástól függetlenül alkalmazunk transzformációs függvényeket. Ilyenkor természetesen más-más függvényeket használhatunk csatornánként, így lehetőség nyílik a képen a színek viszonyának, relatív dominanciájának változtatására. Küszöbözés használata esetén bizonyos színeket tudunk kiemelni a képről, így képesek lehetünk egy adott színű objektumot detektálni.

Az intenzitástranszformációs módszerek rendkívül egyszerűek és gyorsak, robusztusságuk azonban hagy némi kivetnivalót maga után. Éppen ezért gyakorta alkalmazunk képhisztogramra alapuló transzformációkat. A hisztogram egy adott képen vagy színcsatornán az egyes intenzitásértékek relatív gyakoriságát adja meg (ilyen értelemben az egyes intenzitások empirikus eloszlásfüggvénye). A hisztogram gyakran a segítségünkre lehet abban, hogy a képkészítés (általában a megvilágítás vagy az exponálás) hibáit detektálhassuk és javíthassuk.

Alulexponált képek esetén a rövid expozíciós idő miatt nem csapódott be elég foton a szenzortömb egyes elemeibe, így a legvilágosabb képpont értéke is meglehetősen kicsi lesz. Ennek eredményeként a kép intenzitástartománya a teljes tartomány alsó felébe lesz összenyomva, ami miatt a kép részletei nehezen láthatók lesznek. Túlexponált képek esetén hasonló jelenség történik, csak a túl hosszú expozíciós idő miatt a sötét pixelek lesznek túl világosak, és a képi információ a tartomány felső részében lesz összenyomva.

Az ilyen jellegű hibákat kezeli a hisztogramkiegyenlítés[8] algoritmusa. Alul- vagy túlexponált képek esetén a kép hisztogramja ugyanis jelentősen eltér az egyenletes eloszlástól. A hisztogramkiegyenlítés algoritmusának az a célja, hogy a ritka intenzitásértékek összevonásával és a gyakoriak mozgatásával az eredményként kapott hisztogram jobban közelítse az egyenletes eloszlást. Fontos megjegyezni, hogy az algoritmus alkalmazása során (az egyes intenzitásértékek összevonása miatt) információt veszítünk, így a módszert csak a kép láthatóságának javítására szoktuk alkalmazni.



7. ábra

A kép hisztogramja kiegyenlítés előtt és után

Forrás: a szerző szerkesztése

8 Russ 2011.

A számítógépes látás során gyakran használjuk ki a szürkeárnyalatos képekben rejlő intenzitásinformáción felül a színes képek által hordozott extra információt is. Számos egyszerű detektáló algoritmus épül színbeli hasonlóságon alapuló keresésre. Itt azonban számos problémába ütközhetünk. Például ahhoz, hogy a színbeli hasonlóságon alapuló keresés megbízhatóan, robusztusan működjön, arra van szükség, hogy a színeket leíró pixelértékek segítségével könnyen ki tudjuk fejezni a színek hasonlóságát.

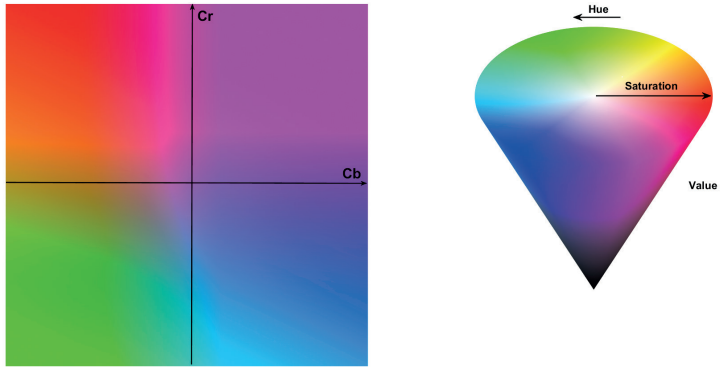
A kamerák által leggyakrabban használt színábrázolás (vagy más néven az RGB-színtér) azonban erre nem alkalmas. Két színt leíró pont geometriai távolsága az RGB-színtérben nem kifejező arra nézve, hogy az emberi érzékelés mennyire érzi hasonlónak a két színt. Ezenfelül a megvilágítási viszonyok megváltozása egy RGB-kép esetén mindhárom értéket módosítja, pedig a képen található objektum színe nem változott.

A színtér-transzformációs eljárások célja, hogy az RGB helyett egy olyan új színreprezentációt adjanak meg, amely információ elveszítése nélkül képes a színbeli hasonlóságot jól leírni, és a megvilágítás változására is robusztus legyen. Ezek a transzformációk ezzel egyben egy új színteret is definiálnak.[9]

Az egyik leggyakrabban használt színtér az YCbCr-család, amelynek több, minimálisan különböző változata van. A színtér három csatornája közül az Y egy elkülönített világosságkomponens, amely az adott szín fényességét reprezentálja. A másik két csatorna, a Cb és a Cr pedig a szín árnyalatát kódolja el. Ezt a színteret gyakorta alkalmazzák digitálisvideó-rendszerekben, valamint a JPEG és az MPEG kódolása során. A gyakorlatban az YCbCr-színteret gyakorta összekeverik az YUV-színtérrel, amely hasonló elven működik, csak éppenséggel analóg rendszerekben használatos.

A másik, képfeldolgozás esetén gyakran használt család a HSV/HSI/HSL-család. E színterek közös jellemzője, hogy a színinformációt két érték, a színárnyalat (hue) és a telítettség (saturation) segítségével írják le, míg a reprezentációk közötti alapvető különbség a fényesség/intenzitás reprezentációjának a módja. Ez a színtér könnyen ábrázolható egy henger vagy kúp formájában. A színalapú feldolgozás esetén rendkívül gyakori mindkét alternatív színtér használata.

[9] Russ 2011.



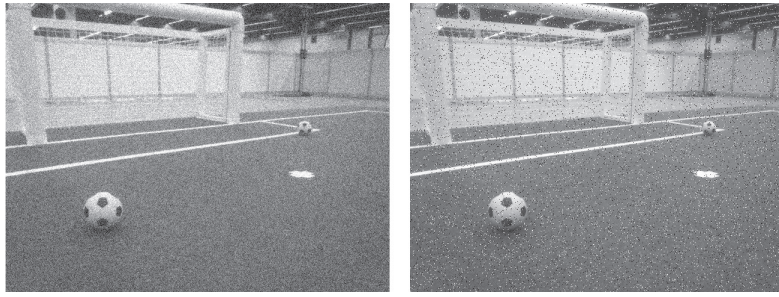
8. ábra

Az YCbCr- és a HVS-színterek ábrázolása

Forrás: a szerző szerkesztése

A valós eszközökkel készített képek mindig zajjal és különböző hibákkal terheltek, amelyek a feldolgozást nehezítik. Ezek a zajok különböző forrásokból származnak, és ettől függően különböző típusai lehetnek. A képeken a leggyakoribb zajtípus a Gauss-zaj, amely a pixelszenzor saját belső zajának és az azt körülvevő elektronika zajának a következménye. Ez a zajtípus tipikusan additív, és pixelenként független.

A másik gyakori zajtípus a só- és borszaj, amely az egyes pixelek értékében számottevő eltérést okoz, de csak ritkán fordul elő, így elszórt, sötét régiókban megjelenő világos pixeleket eredményez (vagy pont fordítva). Ezt a zajtípust leggyakrabban az analóg-digitális átalakító vagy az adásban bekövetkezett bithibák okozzák. Említésre méltó még a digitalizálás során keletkező kvantálási hiba, valamint az esetleges elektromágneses zavarások miatt fellépő periodikus hibák.



9. ábra

Gauss-zaj (balra), illetve só- és borszaj

Forrás: a szerző felvétele és szerkesztése

A képjavító algoritmusok családjának legfontosabb tagjai a különböző szűrő algoritmusok,[10] amelyek a képi hibákat és zajokat hivatottak javítani. Ezek az eljárások konvolúciós szűrésen alapszanak. A képen egy kisméretű szűrőablakkal végighaladva minden egyes pixelpozícióban az adott pixel új értéke a szűrőablak és a pixel lokális környezete között elvégzett konvolúcióművelet eredménye lesz. A konvolúció műveletét az alábbi képlet adja meg:

$$I'(x,y) = \sum_{i=-n}^{n} \sum_{j=-n}^{n} I(x-i, y-j)W(i,j)$$

Ahol I (x, y) az y. képsor x. pixele. Mint a képletből is látható, a konvolúció művelete egyszerűen az adott környezetben lévő pixeleknek a szűrőből vett súlyok alapján számított súlyozott összege. A gyakorlatban mindig olyan szűrőket alkalmazunk, amelyeknél a súlyok összege egy, különben a képet világosabbá vagy sötétebbé tennénk. Fontos megjegyezni, hogy habár a konvolúció képlete alapján a képrészleten és a szűrőn ellentétes irányban kellene haladnunk, a gyakorlatban ezt mégsem így tesszük. Így a valóságban a keresztkorreláció műveletét számoljuk, de ezt mégis konvolúciónak nevezzük, holott a kettő eredménye csak középpontosan szimmetrikus szűrők esetén egyezik meg.

[10] Russ 2011.

A zajszűrésre alkalmazott konvolúciós szűrőket simító szűrőknek nevezzük, amelyeknek legegyszerűbb változata az átlagoló szűrő. Valamivel kifinomultabb változat a Gauss-szűrő, amely a középponttól távolibb pixeleket kisebb súllyal veszi figyelembe, mindezt egy Gauss-haranggörbe alapján. A harangfelület szórásának állításával lehetőség van a simítás erősségének kézben tartására, sőt, ha az egyes irányokban más szórásértékeket adunk meg, akkor létrehozhatunk olyan Gauss-szűrőt, amely az egyik irányban sokkal drasztikusabban simít, mint a másikban.

1	1	1
1	1	1
1	1	1

1	2	1
2	4	2
1	2	1

10. ábra

Az átlagoló (balra) és a Gauss-szűrők (jobbra)

Forrás: a szerző szerkesztése

A konvolúciós simító szűrőknek két problémája van: az egyik, hogy az átlagolás után a zajokat ugyan hatékonyan eltüntetik, azonban a szűrés közben a kép egyes részleteit (főleg az éles váltásokat, éleket) is elmossák, ezzel homályossá téve a képet. Másrészt, mivel az összes ilyen szűrő valamilyen átlagolást végez, ezért az esetleges kiugró értékek (só- és borszaj) ezt az átlagot meglehetősen el fogják téríteni. Ennek eredményeképpen a só és bors jellegű zajokat ezek a szűrők inkább csak elkenik ahelyett, hogy kiküszöbölnék.

Ezekre a problémákra adnak megoldást a rangszűrők. Ezek szintén az adott pixel egy kis környezetét veszik figyelembe, azonban nem a konvolúció műveletét végzik el, hanem ehelyett a környezetben lévő pixeleket intenzitás szerint sorba rendezik, és a sorból egy értéket kiválasztva adnak új értéket az éppen vizsgált képpontnak. A rangszűrők közül a különböző feladatokra maximum, illetve minimum szűrőket szoktak használni; képszűrés esetén a mediánszűrők a legelterjedtebbek.

11. ábra

Az előző zajos képek a Gauss-szűrő (balra) és a mediánszűrő (jobbra) által szűrve

Forrás: a szerző felvétele és szerkesztése

A mediánszűrők az adott pixelt a környezetükben lévő összes pixel intenzitása közül annak mediánjával, vagyis sorba rendezés után a középső értékkel helyettesítik. E szűrés rendkívüli előnye az, hogy az éles határvonalakat, éleket érintetlenül hagyja, míg rendkívül jól szűri a só és bors típusú zajokat. Ennek oka, hogy a mediánstatisztika rendkívül robusztus az átlaggal, a ritka, kiugró értékekkel szemben. A rangszűrők hátránya, hogy a sorba rendezés művelete rendkívül drága a konvolúcióhoz képest, így ezek a szűrők lényegesen lassabb működést eredményeznek. A helyzetet tovább rontja, hogy néhány magasszintű gyorsítási technika (szeparálható szűrők, frekvenciatartománybeli feldolgozás) is csak konvolúciós szűrőkkel végezhető el.

Vákát oldal

3. Lényegkiemelés

A képek számítógépes feldolgozása során az elsődleges feladatunk, hogy a képen olyan jellemző részleteket legyünk képesek megragadni, amelyek később felhasználhatók magasabb szintű feladatok végrehajtására. Ilyen képjellemzőkből számos fajta létezik, amelyek közül a legegyszerűbbet – az egyes képpontok intenzitásértékét – az előző fejezetben tárgyaltuk. A jelenlegi fejezet témája a valamivel bonyolultabb, ebből következően számításigényesebb, azonban robusztusabb jellemzők tárgyalása. Az első alfejezetben a képi élek kinyerését tárgyaljuk, majd ezt követően a képi sarkokat és lokális jellemzőket vizsgáljuk közelebbről. A fejezet végén a bináris képeken található objektumok feldolgozását részletezzük.

3.1. Éldetektálás

A képi élek definíció szerint a képen található szomszédos pixelek között végbemenő nagymértékű, egyirányú intenzitásváltozások. Lényeges tulajdonságuk, hogy az intenzitás csak az egyik irányban változik, míg a másikban konstans, valamint hogy a változás éles, ugrásszerű. A valóságban természetesen a különböző képi hibák, zajok és a véges felbontás miatt a fent leírt ideális élekhez képest a valóságban az átmenet fokozatos, elmosott lesz, valamint lokálisan más irányú változás is elképzelhető.

12. ábra

Ideális élek (balra) és elmosódott, valóságos élek (jobbra)

Forrás: a szerző szerkesztése

A képi élek keresésének legegyszerűbb módja a pixelek egyes irányok szerinti deriváltjának számolása, amelyet a konvolúciós szűréshez hasonló elven tehetünk meg numerikusan. A képen végighaladva minden pozícióban kiszámítjuk az adott pixel és a jobb, illetve alsó szomszédja különbségét, megkapva a kép x, illetve y irányú deriváltjait. A kettő négyzetes összegéből megkapható a teljes derivált nagysága, amely az adott pont élszerűségének mértékeként értelmezhető.

13. ábra

Deriváló szűrő (bal), Gauss-szűrő és a Gauss deriváltja (jobb) egy dimenzióban

Forrás: a szerző szerkesztése

Habár a fent leírt módszer rendkívül gyors és egyszerű, alapvető problémája, hogy a véletlen képi zajok hamis deriváltakat eredményeznek a képen, így a kapott élkép rendkívül zajos lesz. Ez elkerülhető, ha a képet egy Gauss-szűrő segítségével szűrjük, így mérsékelve a zaj hatását. A gyakorlatban azonban az algoritmus gyorsításának érdekében kihasználjuk, hogy mind a deriváló, mind a Gauss-szűrők lineáris műveletek, ezért összevonhatók egyetlen műveletté. Így a két szűrő egymás utáni alkalmazása helyett csak egyetlen szűrést végzünk a Gauss-szűrő deriváltja által meghatározott konvolúciós szűrővel, amely az előző módszerrel megegyező eredményt ad.[11]

A Gauss-szűrő deriváltja mellett elterjedtek továbbá további konvolúciós éldetektáló szűrők, amelyek hasonló elven működnek. Ezek közül a legismertebbek a Prewitt-[12] és a Sobel-operátorok,[13] amelyek irányfüggő éldetektorok. Alkalmazásuk esetén, amennyiben bármilyen irányultságú éleket szeretnénk detektálni, akkor az adott operátor mindkét változatát futtatni kell a képen.

---

11 Russ 2011.

12 Prewitt, J. (1970): Object Enhancement and Extraction. In Linkin, B. S. – Rosenfeld A. eds.: *Picture Processing and Psychopictorics.* New York, Academic Press.

13 Sobel, Irwin (2014): *Isotropic 3x3 Image Gradient Operator.* Presentation at Stanford A.I. Project 1968.

1	0	-1
1	0	-1
1	0	-1

1	0	-1
2	0	-2
1	0	-1

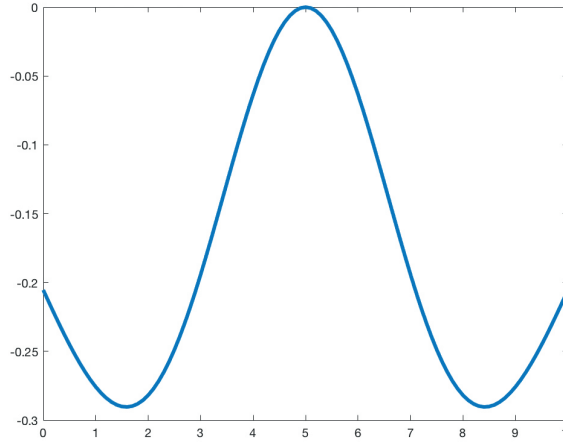
14. ábra

A vízszintes élek deriválására használható Prewitt-operátor (bal) és Sobel-operátor (jobb)

* *Megjegyzés:* A függőleges élekhez használt szűrők ezek transzponáltjai.

Forrás: a szerző szerkesztése

Az első deriválton alapuló szűrők egyik legnagyobb hátránya, hogy a simítás miatt az él homályosan, elkenődve fog látszani, így annak pontos lokalizációja nehézkes. Ez a probléma kiküszöbölhető, ha az élképet még egyszer lederiváljuk, és a deriváltak nullátmenetének pontját keressük meg. Ezt a műveletet természetesen egyetlen lépésben végezzük el egy második derivált konvolúciós szűrő segítségével, amelyet Laplace-szűrőnek nevezünk. A Laplace-szűrőt gyakorta szokták alakja miatt sombrerokalap-szűrőnek is nevezni.



15. ábra

A DoG-szűrő egy dimenzióban

Forrás: a szerző szerkesztése

A gyakorlatban a Laplace-szűrőt néha két eltérő szórású Gauss-szűrő különbségével szokták helyettesíteni. Ezt a megoldást DoG-szűrőnek[14] nevezzük (a DoG az angol Difference of Gaussians kifejezés rövidítése), és számos alkalmazásban használatos. Gyakorta előfordul, hogy a képeket egyszerre több, különböző méretű DoG-szűrő segítségével szeretnénk megszűrni. Ekkor bevett szokás a folyamatot úgy gyorsítani, hogy először külön előállítjuk a különböző Gauss-szűrők által simított képeket, majd magukat a szűrt képeket vonjuk ki egymásból. A kapott eredmény az elvégzett műveletek linearitása miatt megegyezik.

Megfigyelhető, hogy szemben a simító szűrőkkel, ahol minden szűrő elemeinek összege 1 volt, itt minden élkereső szűrő elemeinek összege 0. Léteznek olyan szűrők is, amelyek, habár értékeik elrendezésében inkább az élkereső szűrőkre hasonlítanak (negatív és pozitív értékek más-más oldalon), mégis az értékeik összege 1. Ezeket élesítő szűrőknek[15] nevezzük, és – mint azt nevük is sugallja – képesek a képeken a finom részleteket, változásokat kiemelni, ezzel a képet élesebb érzetűvé tenni. Ezt a szűrőfajtát gyakorta alkalmazzák fotós alkalmazásokban.

Alkalmazás: számos szituációban előfordulhat, hogy egy számunkra fontos objektumról automatikusan kell felvételt készítenünk. A felvételkészítés automatizálása szükséges lehet, amennyiben nem tudjuk kontrollálni, hogy az objektum mikor és hol jelenik meg a képen, vagy olyan nagy mennyiségű felvételt kell készítenünk, hogy a folyamat automatizálására kell szorítkoznunk. Ebben az esetben azonban nem tudjuk garantálni, hogy a kép jól fókuszált lesz, aminek következményeképp a számunkra releváns objektum részletei elmosódottak lesznek, ami a további feldolgozást ellehetetlenítheti.

Ennek elkerülésére használhatunk automatikus fókuszérzékelést,[16] amelynek lényege, hogy – a kép fókuszáltságának mértékét a képből meghatározva – a kamera fókuszát addig módosítjuk, amíg a mérték maximumát produkáló fókuszbeállítást meg nem találtuk. Mivel egy homályos képen a legtöbb képi él elkenődik, ezért a hozzájuk tartozó képi gradiensek nagysága is kisebb lesz. Innen kaphatunk egy kézenfekvő mértéket: minél több olyan pixel van a képen, amelyek helyén a derivált nagysága egy bizonyos

14 Russ 2011.

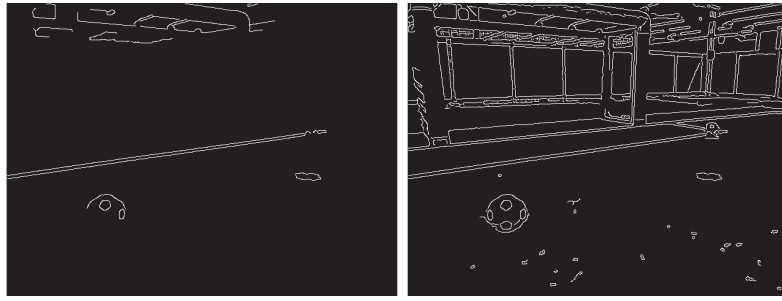
15 Russ 2011.

16 Erteza, Ahmed (1976): Sharpness Index and Its Application to Focus Control. *Applied Optics,* Vol. 15, No. 4. 877–881.

küszöbértéknél nagyobb, a kép annál jobb fókuszú. Fontos megjegyezni, hogy ezzel a módszerrel csak azonos tartalmú képek hasonlíthatók össze.

Az eddigi éldetektáló algoritmusok különböző deriváltszámítási módszereken alapultak, amelyek során megkaptuk minden pixel élszerűségének mértékét. Innentől kezdve azonban a mi feladatunk tervezőként eldönteni, hogy pontosan mekkora határérték felett tekintünk egy képpontot élnek. Ha ezt a hatértéket túlságosan magasra szabjuk meg, akkor előfordulhat, hogy a valós élek kevésbé kontrasztos részeit nem fogjuk tudni detektálni, ha viszont túl alacsonyra tesszük, akkor rengeteg hamis élt fogunk kapni a zajnak és egyéb hatásoknak köszönhetően. A cél persze a kettő közötti kompromisszum megtalálása, azonban ez sem lesz tökéletes.

Ezt a dilemmát igyekszik kezelni az éldetektáló algoritmusok state-of-the-art módszere, a Canny-algoritmus.[17] A Canny-eljárás egy többlépcsős algoritmus, amelynek első lépése, hogy egyszerű deriváló szűrők segítségével kiszámoljuk a képen a vízszintes és a függőleges irányultságú deriváltakat. Ezt követően a kétirányú deriváltakból minden képpontban meghatározzuk a képi gradiens (a legnagyobb intenzitásváltozás iránya) nagyságát és irányát.



16. ábra

A Canny-eljárás eredménye különböző küszöbértékek esetén

Forrás: a szerző szerkesztése

[17] Canny, John (1986): A Computational Approach To Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 8, No. 6. 679–698.

Ezt követően minden egyes élszerű pontból kiindulva, a gradiens irányát követve meghatározzuk a legnagyobb derivált értékkel rendelkező pontot. Az ilyen pontokat meghagyjuk, míg a náluk kisebb derivált értékkel rendelkező szomszédjaikat nullába állítjuk. Ezzel a módszerrel elérjük, hogy a deriváló szűrő használata után kapott homályos élkép pontosan olyan éles legyen, mintha második deriváltat használtunk volna.

Ezt követően a gradiensnagyságokat tartalmazó képet küszöbözzük, azonban egy helyett két különböző küszöbértéket használunk (és így két különböző bináris kép keletkezik). A kisebb küszöbértékkel készített bináris képen nagy valószínűséggel megmarad az összes valódi élpont, azonban számos nem valódi élhez tartozó zaj is keletkezni fog. A nagyobb küszöbértékkel készített bináris képen a valódi élekből biztosan látszani fog valahány képpont, azonban hiányosak lesznek. Cserébe természetesen csak minimális számú zaj fog keletkezni.

A Canny-algoritmus fő erénye, hogy utolsó lépésben e két bináris kép felhasználásával képes egy mindkettőnél lényegesen jobb minőségű élképet létrehozni. Ezt úgy teszi, hogy a nagyobb küszöbértékkel rendelkező, hiányos élképhez hozzáveszi a másik élképről azokat az élpontokat, amelyek szomszédosak a hiányos képen is megtalálható élpontokkal. Ezt iteratívan addig végzi, amíg már nem tud több élpontot hozzáadni a hiányos élképhez. Ennek eredményeképp a Canny-algoritmus fel tudja használni a megengedőbb küszöbértékkel készült élképet arra, hogy a szigorúbb képen keletkező hiányokat betöltse, a valódi élekhez nem kapcsolódó zajokat azonban nem veszi át.

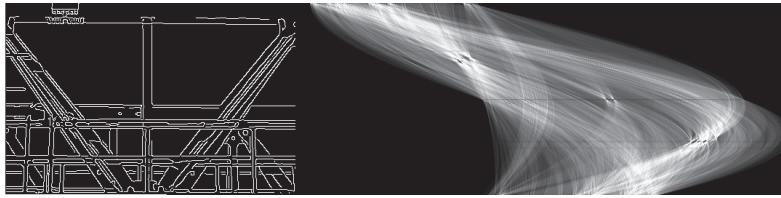
Az éldetektálás során rendkívül gyakran előfordul, hogy különféle egyszerű alakzatok (téglalap, kör) határvonalait keressük, amely esetben célszerű lehet a megtalált élpontokra egy egyenes modellt illeszteni, így a képen megtalált egyenesszerű elrendezésben található pontokat egy paraméteres modellel leírhatjuk, amely az alakzatok detektálását rendkívül megkönnyíti. A probléma nehézsége, hogy természetesen a képen talált élpontok csak egy része fog egyenesekre illeszkedni, és azok közül is számos pont külön-külön egyenesre illeszkedik, így egyszerre kell azt meghatároznunk, hogy mely pontok illeszkednek egy egyenesre, és hogy milyen paraméterek írják le ezt az egyenest. Ha a két kérdés közül bármelyikre tudnánk a választ, a probléma megoldása triviális lenne.

Erre a problémára az egyik legnépszerűbb algoritmus a Hough-transzformációra épülő alakzatdetektálás. A Hough-transzformáció[18] egy koordinátatranszformáció, amely a kép pontjait a megszokott képkoordináta-rendszerből (x, y) az egyenesek paraméterei (r, θ) által kifeszített térbe viszi át. Ebben a térben egy egyenest egyetlen pont (számpár) ír le, amelynek egyik eleme az origóból az egyenesre állított merőleges szakasz hossza (r), a másik eleme pedig ennek a szakasznak az x tengellyel bezárt szöge (θ).

Érdekesebb azonban, hogy a kép egyes pontjai hogyan képződnek le a Hough-térbe. Ezt a leképzést a Hough-transzformáció során úgy végezzük el, hogy az (r, θ) által kifeszített Hough-térben egy képpont képe az összes olyan egyenes lesz, amelyre az adott pont illeszkedik. Habár egy adott pont végtelen számú egyenesre illeszkedik, mégsem illik rá az összes létezőre, így egy konkrét (x, y) pont képét a Hough-térben az alábbi görbe adja meg:

$$x \cos\theta + y \sin\theta = r$$

Vagyis egy képpont képe a Hough-térben egy szinuszgörbe lesz. Ha a Hough-térben két pont görbéje metszi egymást, akkor az azt jelenti, hogy mindkét pont ráillik arra az egyenesre, amelyiket a két görbe metszéspontja ír le (emlékezzünk: a Hough-térben egy pont egy egyenest ír le).



17. ábra
Élkép és a Hough-transzformált
Forrás: a szerző szerkesztése

[18] Duda, Richard O. – Hart, Peter E. (1972): Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM,* Vol. 15, No. 1. 11–15.

Ebből az összefüggésből egy egyszerű következtetés vonható le: ha a Hough-térben sok olyan görbénk van, amelyek megközelítőleg egy pontban metszik egymást, akkor ez azt jelenti, hogy a képtérben sok olyan pont van, amelyek ráillenek ugyanarra az egyenesre. Ebből kiindulva a Hough-féle transzformációs egyenesdetektálás algoritmusa könnyedén adja magát: először transzformáljuk az összes élpontot a Hough-térbe, majd megkeressük azt az egyenest, amelyiken a legtöbb ráillő élpontunk található. Ezt követően ezeket az élpontokat töröljük a Hough-térből, majd újra megkeressük a legtöbb pontra illeszkedő egyenest. Ezt addig ismételjük, amíg találunk olyan egyenest, amelyre legalább egy küszöbértéknyi pont illeszkedik.

Fontos megjegyezni, hogy a Hough-transzformációt nemcsak egyenesek, hanem bármilyen, egyszerű paraméterek által leírható formára el lehet végezni. Különböző típusú Hough-transzformációkat alkalmazva tehát különböző egyszerű alakzatokat tudunk detektálni. Egyenesek mellett gyakorta alkalmazzák a Hough-transzformációt körök, illetve ellipszisek detektálására is. Érdemes megemlíteni az általánosított Hough-transzformációt, amelynek segítségével általános formák is észlelhetők.

Alkalmazás: A Hough-transzformációt előszeretettel alkalmazzák arra, hogy egyenes szakaszokból álló egyszerű formákat detektáljanak a képen.[19] Ez a módszer könnyedén felhasználható például igazolványkártyák automatikus felismerésére, amely az okos közigazgatás egyik legalapvetőbb feladata. Egy igazolványról készült képen a kártya szélei általában markáns, jól kivehető, egyenes éleket okoznak, amelyeket például a Canny-eljárás könnyedén észlel. Ezt követően a Hough-transzformáció segítségével egyenesekké konvertáljuk ezeket, majd megkeressük az ezek által meghatározott, téglalap alakú objektumot.

3.2. Feldolgozás frekvenciatartományban

A számítógépes látás tudományterületén szokványos egy kétdimenziós képet a képsík két dimenziójának függvényeként értelmezni. Ebben a vízszintes x és a függőleges y koordináták által kifeszített térben a kép diszkrét

[19] Fernandes, Leonardo A. – Oliveira, Manuel M. (2008): Real-Time Line Detection Through an Improved Hough Transform Voting Scheme. *Pattern Recognition,* Vol. 41, No. 1. 299–314.

pontokban elhelyezkedő, pontszerű impulzusok összeségeként írható fel, ahol az egyes impulzusok nagyságát az egyes pixelértékek adják meg. Ezt az alábbi képlettel írhatjuk fel:

$$I(x,y) = \sum_{j=0}^{H} \sum_{i=0}^{W} p(x,y)\delta(x-i, y-j) \qquad \delta(x,y) = \begin{cases} 1, ha\ x = y = 0 \\ 0\ egyébként \end{cases}$$

Ahol p (x, y) a pixelérték az y-adik sor x-edik oszlopában, H és W pedig a kép magassága és szélessége. A képeknek (és általánosságban a függvényeknek) azonban nem ez az egyetlen ábrázolási módja. A Fourier-sorfejtés tételének értelmében minden periodikus függvény felírható különböző frekvenciájú szinusz- és koszinuszfüggvények összegeként, ahol minden egyes frekvenciához tartozó függvényhez külön-külön amplitúdó (nagyság) és fázis (eltolás) tartozik. Ezeket a bizonyos frekvenciájú szinusz-koszinusz párokhoz meghatározott amplitúdókat és fázisokat nevezzük a kép spektrumának, a kép ezen értékek által történő megadását pedig a kép frekvenciatartománybeli ábrázolásának.[20]

A kép spektrumát általában a gyors Fourier-transzformáció (FFT-Fast Fourier Transform)[21] algoritmusával szoktuk meghatározni. A Fourier-transzformáció fontos összefüggése, hogy a periodikus jelek spektruma diszkrét lesz, a diszkrét jelek spektruma pedig periodikus. Ez utóbbi számunkra szerencsés, mivel a kép is egy diszkrét jel, így a spektruma periodikus lesz, vagyis elég belőle egyetlen (véges méretű) periódust tárolni, így információ elvesztése nélkül tudjuk a képet frekvenciatartományba, majd visszakonvertálni. A számítógépek fizikai korlátai miatt azonban a kép spektrumát is csak diszkrét függvényként, mintavételezve tudjuk tárolni, így az összes frekvenciatartománybeli műveletünk azt fogja feltételezni, hogy a kép a szélein túl minden irányban periodikusan ismétlődik. Ez a viselkedés megváltoztatja, hogy egyes, egyébként ekvivalens algoritmusok miként működnek annak függvényében, hogy azokat kép- vagy frekvenciatartományban alakítjuk-e át.

Könnyű belátni, hogy ha egy képet fel lehet írni különböző frekvenciájú szinusz- és koszinuszfüggvények összegeként, akkor e periodikus függvények közül az alacsonyabb frekvenciájúak a kép lassabban változó,

---

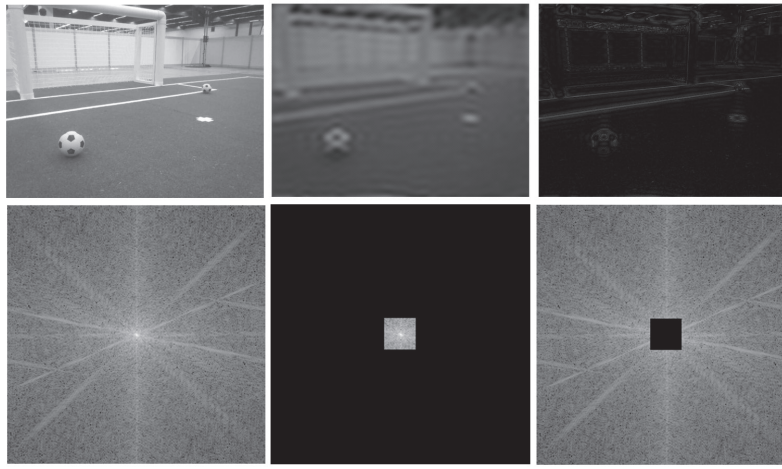
20 Russ 2011.

21 Cooley, James W. – Tukey, John W. (1965): An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation,* Vol. 19, No. 90. 297–301.

simább jellemzőit foglalják magukban, míg a magasabb frekvenciájú komponensek a hirtelen, élesen változó részeket írják le. Ezt az összefüggést több célra is könnyedén kihasználhatjuk: a képi zajok pixelenként függetlenek, tehát hirtelen változásokat okoznak – tehát ha egy kép spektrumában egy aluláteresztő szűrő segítségével ezeket elnyomjuk, akkor zajszűrést végezhetünk. Hasonló módon, a képi élek hirtelen, éles változások az intenzitásfüggvényben, így ha a kép spektrumában egy felüláteresztő szűrővel csak a magasfrekvenciás komponenseket hagyjuk meg, akkor élkeresést végezhetünk. Ha a kép spektrumának a magasfrekvenciás komponenseit erősítjük, de az alacsonyfrekvenciás komponenseket nem töröljük el teljesen, akkor ez az élesítés műveletének felel meg.

A Fourier-tér és a képtér közötti rendkívüli fontos összefüggés, hogy a képtérben elvégzett konvolúció művelete a frekvenciatartományban egy egyszerű elemenkénti (képek esetében pixelenkénti) szorzásra egyszerűsödik. Ez azt jelenti, hogy a különböző konvolúciós szűréseket lényegesen olcsóbb a frekvenciatartományban elvégezni. Ez különösen akkor előnyös, ha egy képen több szűrést is szeretnénk elvégezni, mert akkor a Fourier-transzformációt és annak inverzét csupán egyszer szükséges elvégezni, amelyek számítási költségét az olcsó szűrések megtérítik.

Fontos kiemelni, hogy egy kétdimenziós kép Fourier-transzformáltja szintén egy kétdimenziós tömb lesz, amelyet lehetséges képként ábrázolni. Mivel minden frekvenciakomponenshez két érték tartozik, egy amplitúdó és egy fázis, ezért gyakorta szokás ezek közül csak az amplitúdót ábrázolni, mivel az az ember számára könnyebben értelmezhető információt tartalmaz. Az így megjelenített kép origójában található a nullafrekvenciás, azaz konstans komponens nagysága, ettől a kép szélei felé haladva pedig az egyre nagyobb frekvenciájú komponensek következnek. Fontos megjegyezni, hogy a kapott kép középpontosan szimmetrikus.



18. ábra

Kép és spektruma (bal), aluláteresztő szűrés (közép), felüláteresztő szűrés (jobb)

Forrás: a szerző felvétele és szerkesztése

Mivel a képek kétdimenziós jelek, ezért a képet felépítő periodikus jeleknek irányuk is van, nemcsak frekvenciájuk és fázisuk (ezért is kétdimenziós a Fourier-transzformált). Így a megjelenített amplitúdóspektrum segítségével nemcsak a képen lévő domináns frekvenciákat, hanem azok irányát is megállapíthatjuk. Ez több esetben is rendkívül hasznos lehet. A 2. fejezetben említést tettünk periodikus zajokról, amelyek általában valamilyen elektromágneses interferencia eredményeként keletkeznek a képen. Ezeket a zajokat simító szűrők segítségével nem lehet szűrni. Frekvenciatartományban azonban könnyedén megkereshetjük és kiszűrhetjük a zajért felelős frekvenciát, és abból is elég csak a megfelelő irányút elnyomni.

Alkalmazás: Egy másik felhasználási lehetőség az egyes dokumentumok (különösképp nyomtatott szövegek) irányultságának ellenőrzése. Egy nyomtatott szövegről készült kép esetén ugyanis a sötétebb sorok és világos sorközök egyenletes váltakozása egy jól kimutatható domináns frekvenciát fog létrehozni a kép spektrumában. E frekvenciakomponens irányát megfigyelve ellenőrizni tudjuk, hogy a szöveg vízszintesen látszik-e a képen,

ami nagymértékben megkönnyíti például az optikai karakterfelismerő algoritmusok pontosságát.[22]

3.3. Képi sarkok, lokális képjellemzők

Az aktuális fejezetben a számítógépes látás egyik alapfeladatát, vagyis a képeket jellemző részletek megragadását kezdtük el tárgyalni. Az első tárgyalt képjellemzők a képi élek voltak, amelyek egyszerű, gyorsan kinyerhető, ellenben mérsékelten robusztus leírói a képeken található objektumoknak. Az élek egyik legfontosabb hiányossága, hogy lokálisan csak egy irányban van változás a képen, így ha az él erre az irányra merőlegesen mozdul el a képen, akkor ezt lehetetlen követni.

Erre a problémára ad megoldást a jelen fejezetben tárgyalt képjellemző, a képi sarok. A képsík olyan pontjait nevezzük így, amelyekből minden irányba kiindulva a kép intenzitásértékében számottevő változást figyelhetünk meg. Ebből a definícióból következik, hogy bármerre is mozduljon el a képen a sarkot tartalmazó objektum, a sarok elmozdulását követni tudjuk, így ezt a számunkra valamilyen szempontból releváns objektumot is követhetjük, detektálhatjuk.

Képi sarkok detektálására az egyik legelterjedtebb módszer a Kanade–Lucas–Tomasi-, vagyis a KLT-sarokdetektor. Működésének alapelve, hogy mivel olyan képrészleteket keresünk, amelyek minden irányban jelentősen változnak, ezért egyszerűen az éppen vizsgált képpont egy adott környezetét minden irányban elmozgatjuk a képen, és az elmozgatott és az adott helyen található eredeti képrészletek között négyzetes eltérést számolunk. Ha ez az eltérés minden irányban nagy, akkor sarokszerű képrészletet találtunk.[23]

A KLT-detektor a gyakorlatban természetesen nem így dolgozik, mivel e művelet elvégzése minden képpontban rendkívül költséges volna. Ehelyett a KLT-detektor kiszámolja a kép x és y irányú deriváltjait, majd

---

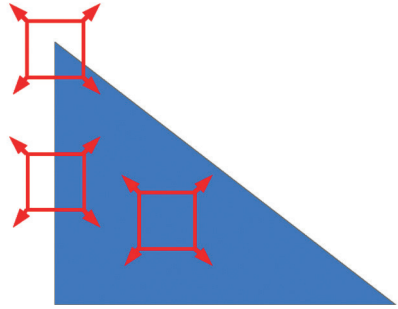
22 Kaur, Mandip – Jindal, Simpel (2013): An Integrated Skew Detection and Correction Using Fast Fourier Transform and DCT. *International Journal of Scientific & Technology Research,* Vol. 2, No. 12. 164–169.

23 Tomasi, Carlo – Kanade, Takeo (2004): Detection and Tracking of Point Features. *Pattern Recognition,* Vol. 37, 165–168.

minden képpont esetében annak az n × n-es környezetében található deriváltakat egy $n^2 \times 2$-es mátrixba rendezi. E mátrix segítségével az alábbi műveletet végzi el:

$$X = \begin{bmatrix} I_x^1 & I_y^1 \\ \vdots & \vdots \\ I_x^N & I_y^N \end{bmatrix} \qquad H = X^T X$$

Ahol I_x^1 és I_y^1 a környezet első pixelének (a végeredmény szempontjából a pixelek sorrendje irreleváns) x és y irányú deriváltjai, H pedig az adott képpont úgynevezett lokális struktúramátrixa. Fontos megjegyezni, hogy amennyiben feltételezzük, hogy a képen az egyes deriváltak várható értéke nulla (ami egy teljesen meglapozott feltételezés, hiszen a deriváltak pont ugyanolyan valószínűséggel negatívak, mint pozitívak), akkor a lokális struktúramátrix a képrészlet deriváltjainak kovarianciamátrixával arányos.



19. ábra
A sarokdetektálás elve
Forrás: a szerző szerkesztése

Fontos azonosság, hogy a lokális struktúramátrix két sajátvektora azt a két irányt fogja meghatározni a képsíkon, amely irányokba a kép a leginkább, illetve a legkevésbé változik. E legnagyobb és legkisebb változások mértékét pedig az egyes sajátvektorokhoz tartozó sajátértékek adják meg. Ez az összefüggés rendkívül jó módszert kínál nekünk a sarkok detektálására: azok a pontok, ahol a legkisebb változás is még elég nagy (vagyis a lokális struktúramátrix legkisebb sajátértéke is nagy) sarokpontoknak tekinthetők.

A KLT-detektor működése során a kép minden eleméhez elvégzi a lokális struktúramátrix számítását, és annak kisebbik sajátértéke alapján hozzárendel egy saroksági mércét, amely alapján a sarokpont pontosan lokalizálható a képen. Fontos megjegyezni, hogy a sarokpont környékén található pixelek sarokságértéke mind nagy lesz, hiszen a képi sarkok nem tökéletesen pontszerű jelenségek. Egyszerű megoldás lehet minden esetben a sarokságértékek lokális maximumait tekinteni a sarokpont helyzetének.

A gyakorlatban szokás a KLT-detektor helyett egy másik eljárást, a Harris-detektort használni, amely szintén a lokális struktúramátrixot használja fel a sarokpontok detektálására, a saroksági mértéket azonban az alábbi képlet alapján határozza meg:

$$R = \det(H) - k * trace(H)$$

Ennek a számolási módszernek hatalmas előnye, hogy lényegesen gyorsabb a sajátértékek meghatározásánál. A Harris-sarokság mellesleg alkalmas élszerű képrészletek detektálására is, ugyanis ilyen jellegű képrészletek esetén a sarokság értéke egy nagy negatív szám lesz. Lényeges további különbség a két detektor között, hogy habár lassabb, a KLT-detektor eredménye közelebb áll az emberi érzékeléshez.[24]

Képjellemzők tárgyalásánál rendkívül fontos megemlíteni az e jellemzőkkel szembe állított robusztussági követelményeket. Célunk ezekkel a jellemzőkkel az, hogy ha ugyanazt az objektumot több képen is látjuk, de esetleg a két kép között különböző transzformációk történtek, akkor e transzformációk közül a képjellemzők minél többre legyenek invariánsak, mivel így ugyanazt a tárgyat különböző képeken is könnyen észlelhetjük.

A legalapvetőbb képtranszformáció az intenzitásváltozás, amely leggyakrabban a megvilágítás megváltozásának köszönhető. Ennek két típusa létezik. Az egyik az additív változás, amely során a képpontok intenzitásértékéhez egy konstans adódik hozzá. Másik típusa a multiplikatív változás, amelynek során a képpontok intenzitása egy konstanssal szorzódik. További gyakori transzformáció a forgatás, valamint az objektum skálájának változása, amely a különböző nézőpontból és távolságból készült felvételek miatt keletkezik. A nézőpontváltozás okoz még perspektív torzítást is a képeken,

[24] Harris, Chris – Stephens, Mike (1988): *A Combined Corner and Edge Detector.* Proceedings of the 4th Alvey Vision Conference.

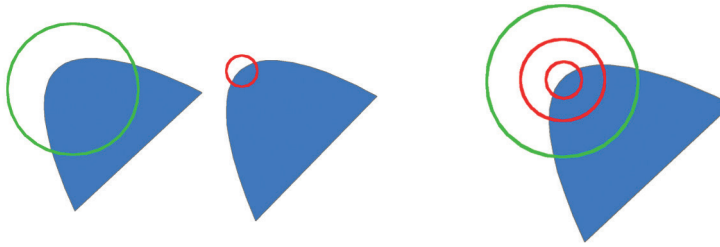
amely legjobban a korábban párhuzamos egyenesek összefutásából figyelhető meg.

A KLT-, illetve a Harris-operátorok e transzformációk közül sajnálatos módon csak kettőre invariánsak: egyrészt az additív intenzitásváltozásra, aminek az az oka, hogy a kép deriváltjai a konstans tagot kiejtik, valamint az elforgatásra, mivelhogy az elforgatás egy mátrix sajátértékeit nem befolyásolja. Multiplikatív intenzitásváltozás esetén a deriváltak is konstanssal szorzódnak, így a sarokságértékek eszerint fognak megváltozni. A skálázás hatásaként az egyik képen sarokszerűnek látszó objektum felnagyítva egy lekerekített élszerű objektumba mehet át, így a skálázás rendkívüli módon befolyásolja a sarokdetektálás eredményét.

Ezt a problémát igyekszik kezelni a skálainvariáns képjellemző transzformáció, vagyis a SIFT-algoritmus (Scale Invariant Feature Transform),[25] amely a perspektív torzítás kivételével minden transzformációra invariáns, így robusztus lokális régióleírót produkál, amelyet széles körben alkalmaznak. Alapelve, hogy sarokszerű pontokat keres a képen, azonban ezekhez nem egyetlen mértéket, hanem a sarokpontok lokális környezetét invariáns módon leíró kódot készít, amelynek segítségével más képeken megtalált jellemzőkkel összevethető, párosítható.

A SIFT-algoritmus első számú alapelve, hogy a sarokdetektálást nem egy, hanem több skálafaktor mentén végzi el, és minden jellemzőhöz egy skálaváltozót rendel hozzá, amelyet a leírókód készítésekor felhasznál, a skálainvarianciát ilyen módon biztosítva. A sarokdetektálást a módszer a 3. fejezetben említett DoG-szűrők (Difference of Gaussians) segítségével végzi el. Korábban ezt a szűrőfajtát éldetektorként ismertük meg, a DoG-szűrő válasza azonban impulzusszerű kitüremkedések és bemélyedések esetén lesz a lehető legnagyobb. Fontos megjegyezni, hogy ha egy adott impulzusszerű képrészleten különböző méretű DoG-szűrőket futtatunk végig, akkor annak a szűrőnek lesz a lehető legnagyobb a válasza, amelynek a mérete a leginkább egybeesik az impulzus szélességével.

[25] Lowe, David G. (2004): Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision,* Vol. 60, No. 2, 91–110.



20. ábra
A skálafaktor hatása a sarokságra (bal), a SIFT sarokdetektálási elve (jobb)
Forrás: a szerző szerkesztése

A SIFT-algoritmus első lépéseként számos, különböző méretű DoG-szűrőt futtat végig a képen, amelyek válaszát eltárolja. Ezt követően megkeresi szűrőválaszok lokális maximumait, de nem csak a kép x és y koordinátája, hanem a szűrők mérete szerint is. A legnagyobb válasszal rendelkező síkbeli koordináta lesz a sarokpont helyzete, a maximális válaszú szűrő mérete pedig az ehhez hozzárendelt skálafaktor. Érdemes tudni, hogy a SIFT nem elégszik meg a diszkrét pixelek és szűrőméretek által kvantált maximumpozícióval, hanem a válaszértékekre lokálisan egy polinomot illeszt, és ennek a maximumhelyét keresi meg. Ezzel a módszerrel a sarokpont helyzetét szubpixeles (vagyis pixelméret alatti) pontossággal képes meghatározni, amely számos alkalmazás esetében követelmény lehet.

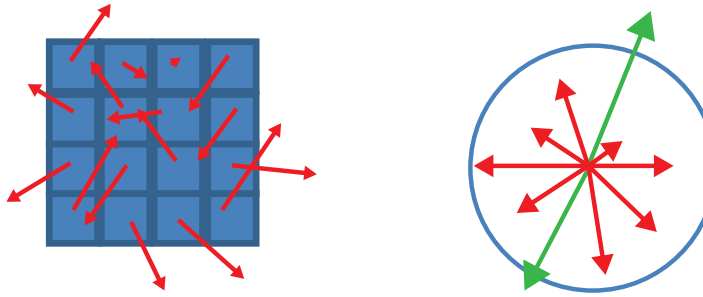
Érdemes megjegyezni, hogy az eljárás a szűrés folyamatát két módon is gyorsítja: egyrészt a DoG-szűrők futtatása helyett az algoritmus különböző méretű Gauss-szűrőket futtat a képeken, majd ezeket vonja ki egymásból, így gyorsítva a működést. Másrészt, amikor az éppen aktuális Gauss-szűrő mérete eléri az eredeti kétszeresét, akkor inkább az eredeti szűrőt futtatja a kép feleakkora felbontású változatán, így ugyanazt az eredményt kapja hozzávetőlegesen negyedannyi számítás segítségével.

A SIFT-algoritmus második lépése a megtalált sarokponthoz tartozó leírókód generálása. A SIFT minden sarokponthoz egy 128 számból álló kódot generál, amely a sarokpont 16 × 16 pixeles környezetének kinézetét írja le. Ezt a 16 × 16 pixeles környezetet mindig az adott sarokpont skálája szerint átméretezett képből vesszük, így biztosíthatjuk a leírókód skálainvarianciáját. Mivel a leírókódot nem közvetlenül a környezet intenzitásértékeiből, hanem annak gradienseiből (az x és y irányú deriváltak által

alkotott kétdimenziós vektorból) készítjük, így a KLT-eljáráshoz hasonlóan az additív intenzitásváltozásra való invarianciákat is biztosítjuk.

A SIFT-algoritmus egyik legforradalmibb ötlete az elforgatásinvariancia biztosítása. Ennek elvégzéséhez a pont környezetének gradienseihez hisztogramot készítünk, amely azt ábrázolja, hogy az egyes irányokba mekkora gradiensek találhatók ebben a környezetben. A gradienshisztogram készítése során a kép gradienseit 36 irány közül osztjuk be valamelyikbe, így a kört 10 fokos intervallumokra osztjuk. Az intervallumhatárok felé mutató gradienseket egyenlően elosztjuk a szomszédos rekeszek között. Fontos részlet, hogy a sarokponttól távoli képpontok gradienseit kisebb súllyal vesszük figyelembe.

A kapott gradienshisztogram-maximum helyét (vagyis azt az irányt, amelybe a leginkább változik a kép ebben a környezetben) megkeressük, és azt az irányt tesszük meg az adott sarokpont orientációjának. A képet ezt követően úgy forgatjuk el, hogy a sarokpont orientációja egy előre meghatározott irányba (például függőlegesen felfelé) mutasson, és a végső leírókódot az elforgatott (és korábban átskálázott) képen található 16 × 16 pixeles környezetből készítjük, így biztosítva annak invarianciáját az elforgatásra. Fontos megjegyezni, hogy ha a gradienshisztogram maximális értéke nem egyértelmű, akkor minden, a maximálishoz közel lévő orientációhoz készítünk egy külön leírókódot.



21. ábra

A 4 × 4-es környezet gradiensei és az azokból számolt hisztogram

Forrás: a szerző szerkesztése

A leírókód készítéséhez a felhasználandó 16 × 16 pixeles környezetet 16 darab 4 × 4 pixeles részre osztjuk, és ezekhez külön-külön, a fent leírt

módon gradienshisztogramokat készítünk. Az egyetlen különbség, hogy az egyes gradienshisztogramok csak 8 és nem 36 rekeszből állnak, így a felbontásuk 45 fok. Az így kapott hisztogramok értékeit egymás után egy vektorba írjuk, így összesen 128 számot kapunk. Ezt a kapott vektort normáljuk úgy, hogy a benne szereplő számok négyzetösszege egy legyen. Ezzel a normálással elérjük, hogy a multiplikatív intenzitásváltozás, amely a gradienseket is konstanssal szorozza, a végső leírókódot ne tudja befolyásolni.

A SIFT-eljárás végére egy olyan jellemződetektálási és -leírási módszert kaptunk, amely a fent említett transzformációkra invariáns. A módszer hátránya, hogy rendkívül számításigényes, modern asztali számítógépeken sem képes valós időben futni. Az algoritmus publikálása óta azonban készült számos más eljárás, amelyek mind a SIFT alapötletére építenek, de lényegesen kevesebb számítást igényelnek. Ezek közül kiemelendő a SURF-módszer,[26] amely hasonló robusztusság mellett kiválóan párhuzamosítható grafikus kártyák segítségével, valamint az ORB,[27] amely az invarianciák megtartása mellett tud valós időben futni.

Alkalmazás: A lokális képjellemzőknek számos alkalmazási területe létezik. Használatosak például képek illesztésére, ami panorámakészítésnél vagy a következő kötetben tárgyalt 3D-látásnál rendkívül fontos. Az egyik legkézenfekvőbb alkalmazás azonban merev objektumok felismerése referenciakép alapján. Ebben az esetben olyan tárgyakat kívánunk felismerni és lokalizálni, amelyek nem deformálódnak, valamint nem létezik túlságosan nagyszámú variáció belőlük. Ekkor a lokális gradienshisztogram-alapú képjellemzőket a referenciaképen megkeressük, majd az aktuális képen található jellemzőket ezekkel összevetjük, a kódok hasonlósága alapján párosítjuk. Nagyszámú egyezés esetén a keresett objektumot megtaláltuk.

Ezt a módszert gyakran alkalmazzák akkor, ha olyan objektumokat kell megkeresni, amelyeket a rendszer tervezője már nem tud befolyásolni, vagyis nem tudjuk könnyebben felismerhetővé tenni. Gyakran alkalmaznak ilyen módszereket valós tárgyakat is felhasználó virtuális- vagy kiterjesztettvalóság-rendszerek, valamint különböző kameraalapú

---

[26] Bay, Herbert – Ess, Andreas – Tuytelaars, Tinne – Gool, Luc Van (2008): SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding,* Vol. 110, No. 3. 346–359.

[27] Rublee, Ethan – Rabaud, Vincent – Konolige, Kurt – Bradski, Gary (2011): *ORB: An Efficient Alternative to SIFT or SURF.* IEEE International Conference on Computer Vision.

megfigyelő- és követőrendszerek (például tér- és forgalommegfigyelő kamerás rendszerek).

3.4. Bináris képek feldolgozása

Már említettük korábban, hogy a képeknek több fajtája létezik. Ezek közül a leggyakoribbak az egycsatornás szürkeárnyalatos, illetve a háromcsatornás színes képek. Szintén gyakran előfordulnak azonban kétállapotú, bináris képek, például éldetektálás esetén, ahol az élekhez tartozó pixelek egyes értéket vettek fel, míg a többi nullást. Ilyen bináris képek azonban számos más művelet eredményeképp is előállhatnak, például küszöbözés, színdetektálás vagy komplex objektumdetektáló eljárások során.

A jelenlegi alfejezetben olyan bináris képeket fogunk vizsgálni, amelyeknek a két állapota közül az egyik az „1”, ami az adott alkalmazás szempontjából relevánsnak tekintett objektumainkat jelöli, míg a „0” a számunkra irreleváns hátteret reprezentálja. Fontos megjegyezni, hogy a megjelenítés során a láthatóság kedvéért a bináris képek képállapotát a maximális fehér és a minimális fekete színek szokták jelölni, arra viszont nincs egyértelmű konvenció, hogy a két szín közül melyik jelöli az objektumot, és melyik a hátteret. A jelen műben következetesen a fehér szín fogja az objektumot jelölni, de ez más forrásokban lehet fordítva is.

A gyakorlatban bármilyen kifinomult módszert is használunk az objektumok elkülönítésére, ez nem fog tökéletesen sikerülni, így – ahogy a színes és szürkeárnyalatos képek esetében is – szükség van a bináris képek javítására. Természetesen mivel a bináris képek hibái más jellegűek, ezért az azok javítására használt módszerek is jelentősen eltérnek. A bináris képeknek alapvetően két jellemző hibája fordul elő: az egyik az olyan pixelek jelenléte, amelyek a valóságban a háttérhez tartoznak, azonban mégis objektumként lettek címkézve, valamint ennek az ellentéte: a tévesen háttérként címkézett objektumpixelek. Az előbbi hibák miatt hamis objektumokat láthatunk a képen, vagy a valóságban elkülönülő objektumokat tévesen összenövesszthetünk. Az utóbbi miatt előfordulhat, hogy lukakat kapunk egyes objektumokon belül, vagy egy a valóságban egybefüggő objektumot tévesen szétválasztunk.

A fenti két bináris képi hibára használható az erózió és a dilatáció művelete. Mindkettő elvégzéséhez egy strukturáló elem definiálására van szükségünk. A strukturáló elem egy tetszőleges alakú ablak, amelyet a konvolúció

műveletéhez hasonló módon minden egyes pozícióban ráillesztünk a képre, és valamilyen műveletet végzünk el a segítségével. A konvolúciós kernelekkel szemben azonban a strukturáló elem értékei csak „0”, „1”, illetve esetenként meghatározatlan („Don’t care”) értékek lehetnek.

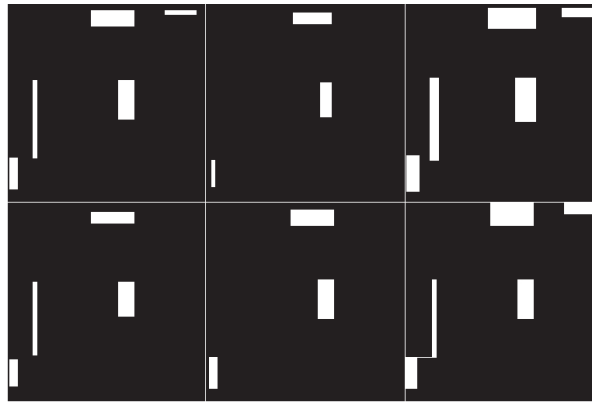
Az erózió művelete esetében az eredménykép adott pixelébe akkor írunk „1” értéket, amennyiben a strukturáló elem tökéletesen ráillik a bemeneti képre az adott pozícióban, vagyis *minden pixelük értéke megegyezik.* Ezzel szemben a dilatáció műveleténél a kimenet akkor lesz „1”, ha a strukturáló elem és a kép *legalább egy pozícióban megegyeznek.* Amennyiben az erózió műveletét egy csupa „1” strukturáló elemmel végezzük, akkor a művelet során az objektumok szélei nullába állítódnak, így az objektumok mérete csökkenni fog. A dilatáció elvégzése esetén ennek a fordítottja történik, vagyis az egyes objektumok nőni fognak.[28]

Fontos megjegyezni, hogy ezeket a műveleteket tetszőleges strukturáló elemmel is el lehet végezni, amely esetben különböző speciális szűréseket tudunk végezni a bináris képen. Egy keskeny, téglalap alakú strukturáló elem segítségével például eltüntethetők a bináris képről olyan élszerű objektumok, amelyek iránya a strukturáló elem hosszabbik oldalára merőleges, azonban az elemmel párhuzamos élek megmaradnak. Hasonló módon egy speciális formájú strukturáló elem segítségével a képről kiszűrhető az összes olyan objektum, amelyek formája nem egyezik meg a strukturáló elemével. Ez a módszer felhasználható például sarkok kiemelésére a bináris képen.

Az erózió és dilatáció műveletének nagy hátránya, hogy az objektumok méretét csökkentik/növelik, így a műveletek alkalmazása után végzett méréseink nem lesznek pontosak. Épp ezért a gyakorlatban sosem szoktuk ezeket az eljárásokat önmagukban alkalmazni, hanem ezeknek kombinációit használjuk. A két leggyakrabban használt eljárás a nyitás és a zárás művelete. A nyitás művelete során először előre meghatározott számú eróziót végzünk el, amely eltünteti a kisméretű, zajszerű objektumokat, majd ezt követően ugyanannyi számú dilatációt csinálunk, amely visszanöveszti a megmaradt objektumokat az eredeti méretükre. Fontos megjegyezni, hogy a nyitásban használt dilatáció olvasztásmentes, vagyis csak akkor állítunk „1” értékben egy korábban „0” értékű képpontot, ha az nem változtatja meg a független komponensek számát. Így azt is el tudjuk érni, hogy a nyitás a kismértékben összenőtt objektumokat szétválassza.

[28] Russ 2011.

A zárás művelete a nyitással ellentétes. Először egy meghatározott számú dilatációt végzünk, amelynek segítségével az objektumokon belül keletkező lukakat betömjük, majd ezt követően ugyanannyi erózió segítségével visszaállítjuk az objektumok eredeti méretét. A két műveletet természetesen egymás után is lehet végezni, így mind a két típusú képhibát javítani tudjuk.



22. ábra

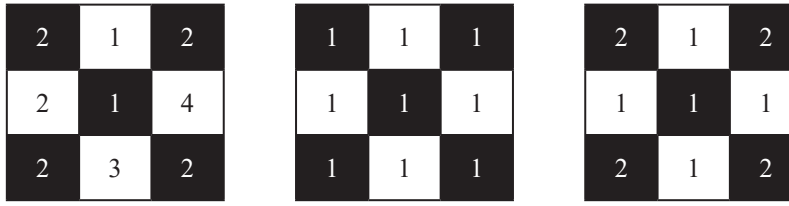
Bináris morfológiai műveletek sorban: eredeti, erózió, dilatáció, erózió téglalap alakú elemmel, nyitás, zárás

Forrás: a szerző szerkesztése

A bináris kép javítása után immáron rendelkezésünkre áll egy meglehetősen jó minőségű kép a számunkra releváns objektumokról, amelyeken különböző méréseket végezhetünk. Mielőtt azonban ezt megtehetnénk, érdemes szót ejteni a pixelek szomszédosságának a kérdéséről. Annak eldöntésére, hogy két pixel szomszédos-e, alapvetően két egyszerű konvenció létezik: a 4, illetve a 8 szomszédosság. A 4 szomszédossági konvenció használata esetén minden pixelnek négy szomszédja van, ezek két oldalt mellette, illetve alatta és fölötte helyezkednek el. A 8 szomszédosság használata esetén az előbbi négy szomszédon felül minden képpontnak további négy szomszédja van, amelyek tőle átlósan helyezkednek el.[29]

[29] Russ 2011.

Bármelyik szomszédosságkonvenciót is használjuk, az sérteni fogja a kép síkjának az úgynevezett Jordan-tulajdonságát, amely azt mondja ki, hogy egy síkot egy összefüggő, zárt görbe pontosan két részre oszt szét. Ez a tulajdonság az ember számára magától értetődő, és célszerű lenne, ha a kép síkjában is teljesülne. Azonban az alábbi ábrák közül az elsőn látható, hogy 4 szomszédság használata esetén az adott konfigurációban a fehér színű pixelek egymással nem szomszédosak, mégis a feketével jelölt háttérterületet két részre osztották. A középső ábrán láthatjuk, hogy 8 szomszédosság használata esetén a fehér pixelek egyetlen összefüggő, zárt görbét alkotnak, azonban a háttér pixelei is egy összefüggő részben maradnak. Ahhoz, hogy a sík Jordan-tulajdonságát megtarthassuk az objektum és a háttér közül az egyikhez 4, a másikhoz pedig 8 szomszédságot kell használnunk. A jobb oldali ábra esetében a fehér előtérobjektumhoz 8-szomszédságot használunk, így ezek a pixelek egy zárt görbét alkotnak, a háttérhez azonban 4-szomszédságot, így a középen lévő pixel nem lesz szomszédos egyik fekete pixellel sem, így a háttér valóban két részre oszlik. Az ábrák esetében feltételezzük, hogy az ablak határain túl a háttér folytatódik a végtelenségig, így a sarokokban található fekete pixelek mindig ugyanahhoz az összefüggő részhez tartoznak.



23. ábra

A Jordan-tulajdonság teljesülése: a pixelekben szereplő számok az összefüggő előtér-, illetve háttérobjektumok sorszámai

Forrás: a szerző szerkesztése

Ennek tisztázása után az objektumok számos tulajdonságát meghatározhatjuk a bináris képen. Ezek közül az egyik első az objektumok alakjának, formájának leírása. Ennek egyik módszere a lánckód alapú határábrázolás. A lánckód használata során egy sorszámot rendelünk minden irányhoz [0–3] vagy [0–7] között, a szomszédossági konvenció függvényében. Ezt követően egy adott kiindulási pontból sorban végig haladunk az objektum határának

összes pontján, minden lépésnél feljegyezve annak irányát. A kiindulási pontba visszaérve megkapjuk az objektum körvonalának egy leíróját.[30]

A lánckód felhasználható az objektum kerületének meghatározására, azonban ebben az esetben figyelembe kell venni azt, hogy (különösen 4 szomszédosság esetén) az objektum körvonalán cikcakkban haladtunk végig, ami mesterségesen megnöveli a kerület hosszát. Éppen ezért az egyes lépések euklideszi távolságát felhasználva sokkal pontosabb becslést kaphatunk. Érdemes megjegyezni, hogy a korábban ismertetett műveleteket felhasználhatjuk arra, hogy előállítsunk egy olyan bináris képet, ahol csak az objektumok határvonalai vesznek fel „1" értéket. Ehhez a képen eróziót kell végeznünk, majd az így keletkezett és az eredeti képet egymásból kivonva megkapjuk az úgynevezett kontúrképet.

Az egyes objektumoknak rendkívül fontos leíró mennyisége még a nyomaték, avagy idegen kifejezéssel a momentum.[31] Nyomatékból számos rend létezik, amelyek közül számunkra a nullad-, illetve az elsőrendű nyomatékok hasznosak. A nulladrendű nyomaték egész egyszerűen a pixelintenzitások összege. Mivel itt bináris képeket tárgyalunk, így az itteni objektumok nulladrendű nyomatéka azok területét fogja megadni. Az objektumok tömegközéppontját pedig meghatározhatjuk az első- és a nulladrendű nyomatékok segítségével az alábbi képlet alapján:

$$x = \frac{M_x^1}{M_0}, \quad y = \frac{M_y^1}{M_0} \qquad M_x^1 = \sum x\, I(x,y), \quad M_y^1 = \sum y\, I(x,y), \qquad M_0 = \sum I(x,y)$$

Amennyiben egy objektum kerülete, területe és tömegközéppontja rendelkezésünkre áll, még számos más alakleírót is hozzárendelhetünk az egyes objektumokhoz, amelyek segítségével azonosíthatók, osztályozhatók lesznek. Ilyen leíróból számos létezik; ezekre az egyik legegyszerűbb példa az objektum kerületének és területének az aránya.

Elterjedt megoldás még a kontúr lenyomatának használata, amelyet úgy számolhatunk ki, hogy egy bizonyos irányból elindulva minden irányban megmérjük az objektum középpontjának és a határvonalának a távolságát, és az így kapott függvényt használjuk az objektum alakjának leírójaként. A kezdőpontot általában úgy választjuk meg, hogy következetesen a legnagyobb távolságú irányból indulunk ki, így elforgatás esetén is

[30] Freeman, Herbert (1961): On the Encoding of Arbitrary Geometric Configurations. *IRE Transactions on Electronic Computers,* Vol. 10, No. 2. 260–268.

[31] Russ 2011.

ugyanazt a leírót kapjuk. Amennyiben a kapott leíró függvényt normáljuk, akkor a skálázásra is invariáns leírót kaphatunk.

Rendkívül elterjedt leíró mérték még az Euler-szám, amely az egy objektumban megtalálható összefüggő régiók számának és az objektumon belül található lyukak számának a különbsége. Ez a mérték rendkívül jól használható olyan esetekben, amikor az objektumok formája számottevő torzulásnak lehet kitéve, azonban a legtöbb gyakorta előforduló geometriai torzítás ezt a tulajdonságot nem változtatja meg, így a különböző Euler-számú objektumok megkülönböztethetők maradnak.[32]

Az utolsó lényeges, bináris képeken végzendő algoritmus az objektumok számlálása és felcímkézése. Ennek az eljárásnak a célja az, hogy az egymástól elkülönülő objektumokat mind egyedi címkével lássuk el, majd a címkézés befejezésekor a legnagyobb címke sorszámából megkapjuk az objektumok számát. A címkézésre alapvetően két elterjedt módszer létezik: a rekurzív, illetve a szekvenciális módszer.

A rekurzív módszer egy rendkívül egyszerű algoritmus.[33] A lépései a következők:

1. Az első címkézetlen „1” pixel megkeresése, és L címkével való megjelölése.
2. A pixel összes „1” értékű szomszédjának L címkével való megjelölése, és a 2. lépés meghívása az összes szomszédra.
   2.1. Ha nincs több jelöletlen „1” értékű szomszéd, akkor leáll az algoritmus.
3. Ugrás az 1. pontba, és az L inkrementálása.

A módszer lépései rendkívül egyszerűek, azonban nagy, összefüggő objektumok esetén a rekurzió rendkívül mélyre mehet, ami problémákat eredményezhet különösen kis teljesítményű feldolgozó eszközök esetében. Ilyenkor célszerű a bonyolultabb, szekvenciális módszert alkalmazni. Ez az algoritmus a kép pixelein sorban halad végig, és minden új „1” értékű képpont esetén az alábbi szabályrendszer alapján ad új értéket a pixelnek:

- Ha a képpontnak csak a felső vagy csak a bal oldali szomszédja címkézett, akkor a jelenlegi pixel az ő címkéjüket kapja.

---

32 Russ 2011.

33 Vincent, Luc – Soille, Pierre (1991): Watersheds in Digital Spaces. An Efficient Algorithm Based on Immersion Simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 13, No. 1. 583–598.

- Ha a felső és a bal oldali szomszéd ugyanazt a címkét viseli, akkor szintén ugyanazt a címkét kapja.
- Ha a felső és a bal oldali szomszéd különböző címkét visel, akkor a pixel a felső képpont címkéjét kapja, valamint egy külön tárolóba feljegyezzük a két címkeérték egyenlőségét.
- Ha a képpontnak nincsen címkézett szomszédja, akkor új címkét vezetünk be a számára.

A szekvenciális algoritmus futásának végén még egyszer végig kell haladnunk a képen, hogy a futás közben feljegyzett, ugyanahhoz az objektumhoz tartozó címkék helyett egy közös címkét adjunk az objektumoknak.[34]

Alkalmazás: A bináris képek feldolgozásának egyik szemléletes alkalmazása a készpénzérmék automatikus számlálása, amely az ilyen fizetőeszközöket alkalmazó tranzakciókat képes gyorsítani.[35] Ehhez célszerű egy külön berendezést készíteni, ahol egy meghatározott színű lapra lehet az érméket lehelyezni, amit egy ismert távolságra lévő számítógéphez kapcsolt kamera figyel. A meghatározott háttér miatt a kép küszöbözéssel könnyedén binárissá tehető úgy, hogy azon az érmék lesznek „1" értékkel jelölve. Az érmék különállóságát és lukmentességét egymás után végzett nyitás- és zárásműveletekkel biztosíthatjuk. Ezt követően címkézés segítségével különválasztjuk az egyes objektumokat, amelyeknek a területe alapján következtetünk az érme címletére. Természetesen az érmék valódiságát szükséges mintaillesztéssel is ellenőrizni.

34 Russ 2011.

35 Cooray, Thilan – Fernando, Shilan (2011): *Visual-Based Automatic Coin Counting System.* SAITM Research Symposium on Engineering Advancements.

Vákát oldal

4. Magas szintű látás

A jelenlegi fejezet témája a magas szintű látás, amelynek alapvető célja, hogy a korábbi fejezetek során bemutatott módszerek segítségével kinyert képi információk alapján a számítógépes rendszer képes legyen döntéseket hozni vagy adott esetben az emberi döntéseket támogatni. Először a magas szintű látás legegyszerűbb alapesetét, az osztályozást mutatjuk be, majd erre építve tárgyaljuk a detektálás és a szegmentálás feladatait. A fejezet végén pedig a mozgókép-feldolgozás területének, vagyis a videoanalitikának a módszereit részletezzük.

4.1. Képosztályozás

A magas szintű számítógépes látás legalapvetőbb feladata a képek osztályozása. Ez akkor szükséges, ha a feladatunk bemenete egy vagy több kép, az elvárt kimenet pedig egy két- vagy többállapotú címke. A címke állapotainak száma alapján beszélhetünk bináris vagy több állapotú osztályozásról. Magának a címkének számunkra szemantikus jelentősége van, így meghatározása által a képen például bizonyos objektumok vagy tevékenységek jelenlétére következtethetünk. Fontos megjegyezni, hogy osztályozás esetén nem következtetünk az objektumok számára, pozíciójára és egyéb tulajdonságaira. Többállapotú osztályozás esetén megkülönböztethetünk kizáró és nem kizáró osztályozást annak függvényében, hogy egy képhez több címke hozzárendelését is megengedjük-e.

Amennyiben magas szintű szemantikus információt szeretnénk a képből kinyerni, számos nehézséggel kell szembenéznünk. Ezek közül az első, hogy ugyanannak az objektumnak a képe különböző megvilágítások miatt jelentős változásokon mehet keresztül, így az objektumot adó pixelek numerikus értéke megközelítőleg sem fog megegyezni. Hasonló nehézségeket eredményez az elforgatás, skálázás és a perspektív torzítás, amelyek ugyanarról az objektumról készített felvételeken ugyanúgy számottevő változásokat okoznak. További problémákkal járhat, hogy egyes

objektumok képesek deformálódni, ami szintúgy megváltoztatja a leíró jellemzők értékét. Valódi képeknél szintén gyakori, hogy az objektumok egy jelentős része takarásban van, így a felismerést csak egy részleges kép alapján tudjuk elvégezni.

Eddig azonban csupán egyetlen objektumról esett szó, a jelenlegi feladat során azonban egy *szemantikus osztályt* szeretnénk felismerni. Egy osztályba számtalan különböző objektum tartozhat, amelyek között (az adott osztálytól függően) jelentős eltérések lehetnek. Sőt, a valós világban gyakoriak az olyan osztályok, amelyek egyes példányai egyáltalán nem mutatnak vizuális hasonlóságot, az azonos osztályba való tartozásukat pedig valamilyen fizikai vagy funkcióbeli hasonlóság alapján tudnánk eldönteni (gondoljunk például különböző kialakítású székekre). Ezt a problémát osztályon belüli variációnak nevezzük, és a szemantikus osztályozása egyik legnagyobb nehézsége.

Általánosságban egy osztályozást végző számítógépeslátás-megoldás számos algoritmus egymás után történő végrehajtásából áll, amelyet algoritmikus csővezetéknek (pipeline) nevezünk. Ezek első lépése a képek készítése és digitalizálása, amelyet egy előfeldolgozó, képjavító (zajszűrések, intenzitástranszformációk) blokk követ. Ezt követően egy jellemző kiemelési fázis következik, amelynek célja, hogy a képen fellelhető információt a pixelintenzitások által meghatározott térből egy ennél nagyobb absztrakciós szinten létező képjellemzők által meghatározott térbe transzformálja. Ezeket a képjellemzőket úgy tervezzük meg, hogy az általuk meghatározott térben könnyen elválaszthassuk a feladat szempontjából releváns információkat a zavaró hatásoktól. Az utolsó lépés egy döntési fázis, amelyben az algoritmus a képjellemzők alapján címkét rendel az adott képhez.

Érdemes a képjellemzők szükségességét egy szemléletes példán keresztül demonstrálni. Lehetséges ugyanis képosztályozást tisztán intenzitás vagy szín alapján végezni, legegyszerűbben például a *k* legközelebbi szomszéd elnevezésű, vagyis a kNN (az angol *k Nearest Neighbours* kifejezésből) algoritmus segítségével. Ennek a végtelenül egyszerű eljárásnak a lényege, hogy egy már ismert címkéjű képekből álló adatbázisban megkeresi az éppen osztályozandó képek *k* darab legközelebbi szomszédját. Ezek a szomszédok aztán többségi elven, szavazással döntik el az új kép címkéjét. A kNN a képek távolságát általában a két kép pixeleinek abszolút vagy négyzetes

különbségeinek összegeként definiálja. A módszerben felhasznált *k* változó értékét a tervező szabadon választhatja.[36]

A megoldás alapvető problémája, hogy az intenzitás és színértékek közti különbségek összege nincs összhangban a képek hasonlóságával, különösen nem a szemantikus osztályok közti különbséggel. Könnyen belátható, hogy a különböző megvilágítással vagy eltérő háttér előtt készült képek különbsége jelentős lesz a rajtuk szereplő objektum osztályától függetlenül. A helyzet különösen rossz olyan objektumok esetén, amelyek hajlamosak számos különböző színezetben előfordulni (például állatok, járművek, ember). E probléma miatt a színinformációt csak olyan esetekben használjuk képek osztályozására, amikor mind az objektumok kinézetét, mind a környezet vizuális tulajdonságait kézben tudjuk tartani. Ilyen szituációkra jó példák a különböző beltéri ipari alkalmazások (például alkatrészfelismerés) vagy a virtuális- és kiterjesztettvalóság-rendszerek.

A képből számos különböző jellemzőt nyerhetünk ki, amelyek különböző robusztusságot eredményeznek eltérő számítási igény mellett. A felhasznált jellemzők megfontolt megválasztása az adott probléma megoldásának minőségét nagymértékben befolyásolja. Mint már említettük, a legkevésbé robosztus jellemzők a szín- és az intenzitásértékek, amelyek előállítása minimális mennyiségű számítást igényel. Ennél nagyobb robusztusságot érhetünk el, ha az élek helyzete és formája (például a Hough-transzformáció segítségével meghatározva) alapján hozunk döntést. A megbízhatóságot tovább növeli, ha a sarkok helyzete alapján próbálunk döntést hozni, a sarokdetektálás számításigénye azonban lényegesen nagyobb. A legrobusztusabb jellemzők egyben a legdrágábbak is: ezek a különböző lokális régióleíró operátorok.

Amennyiben rendelkezésünkre állnak az objektum kontúrjának vagy jellemző pontjainak valamilyen leírói (élek, sarokpontok vagy bináris lenyomat), akkor ezekből további jellemzőket nyerhetünk ki. Ilyen esetekben az osztályozást csupán a képen található formák alapján végezzük el, hiszen a fent felsorolt jellemzők csupán ezt az információt hordozzák. Az ilyen megoldásokat alakfelismerésnek nevezzük. A kétdimenziós képeken található formák leírásának számos módja van, amelyek közül számos módszert már a korábbi fejezetben ismertettünk. Idetartozik a Hough-transzformáció és annak általánosított változata. Bináris képek esetén használhatók a különböző alakleírók, például a nyomatékok, a lenyomat vagy az Euler-szám.

---

[36] Altman, Naomi S. (1992): An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician,* Vol. 46, No. 3. 175–185.

Alakfelismerés esetén gyakorta alkalmaznak még statisztikai leírókat.[37] Ezeknek a lényege, hogy egy egyszerűen számítható geometriai tulajdonságot mintavételeznek a képen, és ennek eloszlását használják az alakzat leírójaként. Például egy kontúrról pontpárokat véletlenszerűen kiválasztva kiszámíthatjuk azok távolságát, majd sok véletlen párt mintavételezve következtethetünk a távolságok eloszlására, ami az adott kontúr formáját leíró alapvető statisztika. Különböző képeken e statisztikák összehasonlításából következtethetünk a képek osztályára.

Az alakfelismerés módszerének két hátránya van. Egyrészt, hogy a deformációk és eltakarások hatásának kezelése meglehetősen nehézkes. Problémás megoldani az osztályon belüli variációk kezelését is. A másik hátrány, hogy a módszer csupán az alakzat tulajdonságait használja fel, annak kinézetét pedig teljesen elveti. Intuitív módon ellenben könnyedén beláthatjuk, hogy a szemantikus kategóriák felismeréséhez mindkét információtípusra szükségünk van.

Erre a problémára jó megoldást nyújthatnak a lokális képjellemzők, amelyek a sarokszerű pontok detektálása mellett egy számos transzformációra invariáns leírókódot is számítottak minden detektált jellemzőhöz. Ezek a jellemzők egyszerre kódolják a kép kinézetét és struktúráját egy kis lokális szeletben, relatív elhelyezkedésükből pedig a globális formára is következtetni lehet. Legnagyobb hátrányuk, hogy számításuk meglehetősen költséges: a korábban bemutatott SIFT-algoritmus futási ideje egy átlagos méretű képre asztali gépen is a másodperces nagyságrendben van.

Lokális gradienshisztogramra épülő képjellemzők alapján történő osztályozásra leggyakrabban vizuális szóhalmazra[38] (angolul: Bag of Visual Words) alapuló eljárásokat szoktak alkalmazni. Ez a típusú eljárás a szöveges dokumentumok osztályozásának tudományterületéről származik. Alapötlete, hogy egy szöveget téma alapján be lehet sorolni a benne előforduló szavak relatív gyakoriságát figyelembe véve. Fontos megjegyezni, hogy a módszer csupán a szavak előfordulását veszi figyelembe, azok sorrendjét, relatív helyzetét viszont nem.

---

37 Floriani, Leila D. – Spagnuolo, Michela (2007): *Shape Analysis and Structuring.* London, Springer.

38 Li, Fei-Fei – Perona, Pietro (2005): *A Bayesian Hierarchical Model for Learning Natural Scene Categories.* IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

24. ábra
A képből kinyerhető vizuális szavak
Forrás: a szerző szerkesztése

A módszer könnyedén átalakítható képek osztályozásának esetére, amennyiben a különböző lokális képjellemzőket vizuális szavakként értelmezzük. Az átalakítás során azonban egy problémába ütközünk: a szóhalmazos osztályozás során rendelkezésünkre állt egy szótár, amely alapján az egyes szavakat le tudtuk kódolni (például az alapján, hogy az adott szó hányadik helyen szerepel a szótárban). A szavak betűnkénti egyezése alapján a szótárban lévő szavakkal meg tudtuk őket feleltetni, valamint egy szinonima-szótár segítségével még az azonos jelentésű szavakat is képesek voltunk azonos kóddal szerepeltetni.

A vizuális szavak esetén azonban nem áll rendelkezésünkre ilyen szótár. A különböző vizuális szavak ráadásul valójában lebegőpontos számok sorozatai, amelyek sosem fognak egymással elemenként megegyezni. Így felmerülhet a kérdés: hogyan tudunk egy képi adatbázisból vizuális szótárt konstruálni, és hogyan tudjuk a képeken talált vizuális szavakat a szótárban szereplő szavak közé beosztani?

Erre a klaszterezés művelete a megoldás, amely a felügyelet nélküli gépi tanulás egyik alapvető módszere.[39] A klaszterezés során a vizuális szóhalmazt úgy osztjuk kisebb csoportokra (klaszterekre), hogy azok minél kompaktabbak legyenek, vagyis a csoport elemeinek a középtől számított távolságainak összege minimális legyen. A klaszterek számának a megválasztása alapvetően a tervező feladata, akinek egy kompromisszumértéket

[39] Forgy, Edward W. (1965): Cluster Analysis of Multivariate Data. Efficiency Versus Interpretability of Classifications. *Biometrics,* Vol. 21, 768–769.

kell találnia a klaszterek kompaktságának csökkentése és a csoportszám túlzott növelése között. A klaszterezés algoritmusairól bővebb információ az 5.3. alfejezetben, valamint a jelen kismonográfia második kötetében található.

Amennyiben a klaszterezés segítségével sikeresen konstruáltunk egy vizuális szótárt, ennek szavaihoz az új lokális képjellemzők már a szóközepektől számított négyzetes hiba minimalizálásával könnyedén beoszthatók. Ha ezt megtettük, az adott képen megtalálható vizuális szavakból egy hisztogramot konstruálhatunk, amely azt fejezi ki, hogy a vizuális szótárban található szavak közül melyik milyen relatív gyakorisággal fordul elő a képen. Ezt a hisztogramot súlyozhatjuk úgy, hogy az egyes szavakat nem keményen rendeljük az egyes szótárbeli szavakhoz, hanem az alapján súlyozva, hogy azok a középponttól milyen távolságra voltak. Ezen a módon a nagy konfidenciával észlelt szavak nagyobb súllyal fognak szerepelni.

Ezt követően a hisztogramot felhasználhatjuk úgy, hogy a képeket bizonyos osztályokhoz rendeljük. A döntés elvégzéséhez számos módszert használhatunk, amelyek általában a gépi tanulás tárházából kerülnek ki. A gépi tanuló algoritmusok általánosságban egy adatbázisból nyerik ki az optimális döntésfüggvényt valamilyen statisztikai vagy optimalizáló módszer segítségével. Az egyik legegyszerűbb példa a tanuló algoritmusra a kNN-módszer, amely ebben az esetben könnyedén alkalmazható. A döntéshez vesszük azt a képi adatbázist, amelyik segítségével a szótárt konstruáltuk, majd ebben megkeressük a jelenlegi kép k legközelebbi szomszédját, ezúttal azonban nem a pixelértékek, hanem a szóhalmazhisztogram alapján számítunk távolságot. Az új kép címkéjét pedig a k legközelebbi szomszéd dönti el többségi szavazás alapján. Fontos megjegyezni, hogy a gépi tanulás megoldásaihoz általában ismernünk kell az adatbázis címkéit.

Alkalmazás: Az osztályozás egyik gyakori alkalmazása biztonsági, illetve térfigyelő kamerák esetén történik.[40] Ilyen alkalmazásoknál gyakran fontos tudni, hogy például egy érzékelt mozgást milyen osztályú objektum okozott, ugyanis ennek függvényében gyakran más döntést lehet hozni. Autópályákon elhelyezett kamerák esetében fontos lehet például az, ha embert detektálunk, mivel ez valamilyen baleset vagy egyéb szokatlan

[40] Varma, Soumya – Sreeraj, M. (2013): *Object Detection and Classification in Surveillance System.* IEEE Recent Advances in Intelligent Computational Systems. Trivandrum, India.

szituáció jele lehet. Hasonló eset, ha egy járműforgalom számára elzárt utcában autót detektálunk. Ugyanígy emberek, járművek és esetleg állatok felismerése beléptető és egyéb biztonsági rendszerek esetén is hasznos.

Egy ettől független alkalmazási terület lehet nagy mennyiségű kép rendszerezése vagy kulcsszavakkal történő ellátása a későbbi könnyebb kereshetőség vagy feldolgozhatóság biztosításának érdekében. Fontos megjegyezni, hogy ilyen esetekben gyakran elegendő, ha a képeket felügyelet nélkül csoportosítjuk. Ez azt jelenti, hogy az algoritmus nem előre ismert kategóriákba próbálja a képet besorolni (például ember, autó, állat), hanem maga az algoritmus definiál kategóriákat valamilyen hasonlósági kritérium alapján. Az egyes kategóriák értelmezését már egy ember fogja elvégezni, ehhez viszont elég minden kategóriából csupán néhány képet megtekinteni.

4.2. Objektumdetektálás

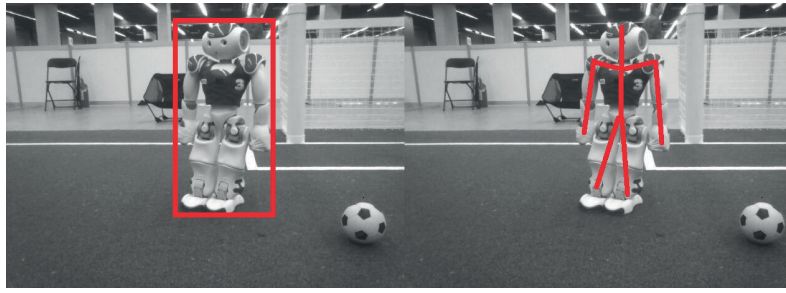
A számítógépes látás alkalmazásai során gyakorta nem elegendő, hogy egy képről általánosságban megmondjuk, hogy milyen osztályba tartozik (vagy hogy milyen osztályú objektumok találhatók rajta), hanem szükséges ezeknek az objektumoknak a pozíciójára is információkat szolgáltatnunk. Az ezekre épülő döntéshozásnál ugyanis gyakran ezen információk alapján teljesen ellenkező akciókat kell végrehajtanunk, az osztály jelenléte önmagában nem hordoz különösebben fontos információt. Szemléltetve: az, hogy egy önvezető autó lát egy gyalogost, alapvetően nem meglepő, hiszen gyalogosok nagy számban fordulnak elő az utak mellett, a pozíció alapján viszont már eldönthetjük, hogy kell-e fékezni.

Az ilyen jellegű feladatoknak több változata van, amelyek egyrészt különböznek aszerint, hogy pontosan milyen pozícióinformációt kívánunk kinyerni. A legegyszerűbb, ha csak az objektum középpontjának koordinátáit szeretnénk a képsíkon megkapni, de ezen felül meghatározhatjuk az objektumot befoglaló téglalapot vagy bizonyos esetekben ennek a minimális területű, elforgatott változatát. Az elforgatott téglalap hosszabbik oldalának irányát választhatjuk az objektum orientációjának. Komplex, deformációra képes objektumok detektálása esetén egy összetett pózt leíró információt is kinyerhetünk, ami pont ezt a deformációt adja meg.

Kezdjük a detektáló algoritmusok tárgyalását az egyszerűbb esetekkel! Ilyenkor olyan objektumok pozícióját keressük, amelyek kinézete jól ismert, lokálisan könnyen leírhatók, és általában merevek. Bár ezek a megkötések

elég szigorúak, számos esetben találkozhatunk ilyen objektumokkal. A különböző virtuális- és kiterjesztettvalóság-rendszerekben alkalmazott mesterséges markerek, alkatrészek és egyéb mesterséges eszközök, valamint a nyomtatott betűk gyakran megfelelnek e követelményeknek.

Ilyen esetekben használható a sablonillesztés[41] (angolul: template matching) algoritmusa. Ez az eljárás rendkívül egyszerű: a detektálandó objektumról először referenciaképet készítünk, majd ezt a mintát a képre minden lehetséges pozícióban ráillesztjük, ezután pedig a kép és a sablon között valamilyen illeszkedési függvényt számolunk. Az illeszkedési függvény szélsőértékeinek pozíciójában pedig detektálást jelzünk.



25. ábra

A befoglaló téglalap (bal) és a póz (jobb)

Forrás: a szerző szerkesztése

A sablonillesztés során alapvetően kétféle illeszkedési függvényt használnak a gyakorlatban. Ezek közül az egyik a sablon és a képrészlet pixelei közötti négyzetes eltérése összege vagy más néven az L2 távolság, amelynek a minimumpontjait keressük. Érdekesebb megoldás azonban a konvolúciós/korrelációs illeszkedés, ahol a sablon és a kép közötti konvolúciót használjuk mérceként. A konvolúció alapvető tulajdonsága, hogy az eredménye akkor lesz abszolút értékben nagy, ha a szűrő és az általa lefedett képrészletre vagy annak inverzére hasonlít. Ennek illusztrálására említhetjük a 4.1. fejezetben bemutatott éldetektáló operátorokat, amelyek valóban úgy néznek ki, mint egy képi él.

A két hasonlósági mérce között lényeges különbség, hogy a konvolúciós megoldás esetén, ha a válasz mindkét szélsőértéke esetén jelzünk detektálást,

41 Russ 2011.

akkor a sablon inverzét is detektáljuk. Ez hasznos lehet például betűk felismerésénél, ahol így egy fehér háttéren a fekete betűs mintát és a sötét háttéren a világos betűket is fel tudjuk ismerni. A sablonillesztési eljárás egyik főbb gyengesége, hogy a forgatásra, skálázásra és a torzításokra rendkívül érzékeny, így ha ezek előfordulnak, akkor minden skálához és orientációhoz külön sablont kell készíteni, és az eljárást az összes sablonnal meg kell ismételni, ami negatívan befolyásolja a sebességet.

Alkalmazás: A mintaillesztés eljárásának az egyik legfontosabb alkalmazása az optikai karakterfelismerés (OCR – Optical Character Recognition) azon esete, ahol nyomtatott karaktereket kell felismerni.[42] Ebben az esetben a korábban ismertetett módszer segítségével, jó irányban beállított szövegre minden nyomtatott karakterhez egy külön mintát végigfuttatva az adott betű elfordulásait lokalizálni tudjuk, és így megkapható a szöveg. Az optikai karakterfelismerés alkalmazási területei pedig végtelenek, kezdve a szkennelt dokumentumok szöveggé konvertálásától a különböző igazolványok automatikus elolvasásáig.

Fontos alkalmazás még a virtuális és kiterjesztett valóság tudományterülete,[43] amelyben gyakorta valósítanak meg beviteli eszközöket oly módon, hogy az eszközökön valamilyen speciális (általában fekete-fehér) jelölő mintázatot helyeznek el. Ezeket a mintákat úgy szokták megválasztani, hogy azok a valóságos objektumokon ne fordulhassanak elő, így e markerek lokalizációja a mintaillesztés segítségével egyszerűen és robusztusan megoldható. Ezt a módszert előszeretettel használják tapintható kiterjesztett valóság rendszerekben, amelyek alapvető elve, hogy a virtuális környezettel való interakciót speciális valós objektumok segítségével valósítják meg.

Amennyiben a detektálandó objektum megjelenésekor komplex, illetve több torzító hatás jelenlétét engedjük meg, akkor kénytelenek leszünk a pixelintenzitásoknál bonyolultabb jellemzőket használni. Erre szerencsére számos, korábban ismertetett megoldás létezik, így ilyen objektumok esetén előszeretettel nyúlunk a lokális, gradienshisztogram-alapú képleírókhoz. Maguk a képleírók ugyanis, amint azt korábban tárgyaltuk,

---

42 Schantz, Herbert F. (1982): *The History of OCR, Optical Character Recognition*. Manchester, Recognition Technologies Users Association.

43 Uchiyama, Hideaki – Marchand, Eric (2012): *Object Detection and Pose Tracking for Augmented Reality. Recent Approaches.* 18th Korea–Japan Joint Workshop on Frontiers of Computer Vision, Kawasaki.

rendkívül robusztusak a legtöbb lényeges képi torzításra, ezért ideálisak az ilyen feladatok elvégzésére.

Lokális képjellemzők alapján történő detektálás esetében feltételezzük, hogy a keresendő objektumról rendelkezünk egy referenciaképpel, amelyet felhasználunk arra, hogy arról a legmarkánsabb lokális képjellemzőket kiemeljük és eltároljuk. Ezt követően az aktuális képen szintén képjellemzőket keresünk, és a referenciaképről elmentett jellemzőkkel párosítjuk. Ezt a párosítást általában a képjellemzők leírókódja közötti különbség alapján tesszük meg: lebegőpontos kódok (mint amilyet a SIFT és a SURF képez) esetén L2 távolságot, míg a bináris kódok (az ORB például ilyet generál) esetén Hamming-távolságot alkalmazunk.

Gyakorta szoktuk a párosításokat egyértelműség és szimmetria alapján is szűrni. Az előbbi azt jelenti, hogy egy adott jellemzőpárosítás estén elvárjuk, hogy a jellemző második legjobb párja legyen lényegesen roszszabb a legjobbnál. Az utóbbi esetén pedig azt várjuk el, hogy ha az adott f_i jellemző legjobb párja a referenciaképről f_r, akkor f_r legjobb párja a kép összes jellemzője közül legyen f_i. Belátható, hogy a jellemzők megfelelő egyedisége és robusztussága esetén a keresett objektum pozíciója környékén a legtöbb jellemzőt megtalálhatjuk.

Természetesen lehetőség nyílik arra is, hogy a párosítások segítségével a referenciaobjektum és a detektált példánya közötti kétdimenziós, merev transzformációt meghatározzuk. Ebben az esetben eltolást, forgatást és irányfüggetlen skálázást engedünk meg, amelynek meghatározásához három lokális képjellemző párosítására van szükségünk. Lehetőség van a két objektum közötti homográfia meghatározására is, amelyben az objektumon már irányfüggetlen skálázás, nyírás és perspektív torzítás is felléphet, ehhez azonban egy negyedik pontpárra is szükségünk van.[44]

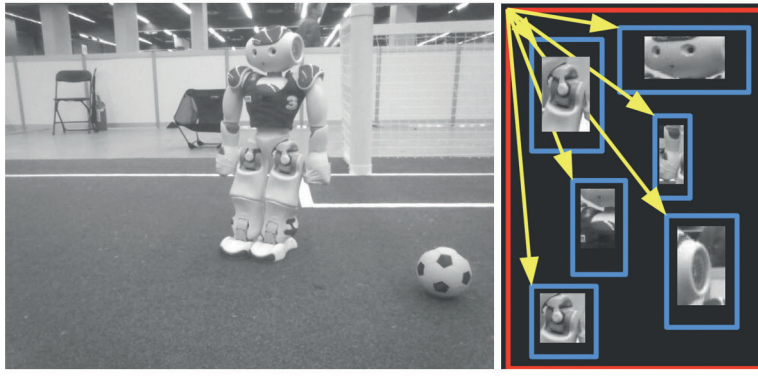
Érdemes kiemelni, hogy amennyiben rendelkezünk az objektumról előzetes, háromdimenziós információval (azaz ismerjük az egyes képjellemzők térbeli elhelyezkedését), akkor 4 pontpár detektálása esetén kétdimenziós képeknél is meg tudjuk határozni az objektum térbeli, hat szabadságfokú (3D-pozíció és 3 tengely körüli forgatás) transzformációját. A transzformációkat általában a legkisebb négyzetek módszerének segítségével becsüljük meg, így célszerű, ha a minimálisan szükséges 3/4 pont

[44] Ghadarghadar, Nastaran – Ataer-Cansizoglu, Esra – Zhang, Peng – Erdogmus, Deniz (2012): *A SIFT-Point Distribution-Based Method for Head Pose Estimation*. IEEE International Workshop on Machine Learning for Signal Processing, Santander.

helyett lényegesen több pontpár áll rendelkezésünkre, mivel ez a becslés pontosságát nagymértékben növeli.

Felmerülhet azonban a kérdés, hogy mi történik, ha egy konkrét objektum helyett inkább egy absztrakt objektumkategóriát, azaz egy osztályt szeretnénk detektálni. Lehetséges-e ilyen esetben a vizuális szóhalmazos módszert alkalmazni? A válasz, hogy alapvetően ugyan lehetséges, a kinyert pozícióinformációk azonban rendkívül pontatlanok lesznek, ugyanis a szóhalmazmódszer semmilyen az egyes vizuális szavak abszolút vagy relatív helyzetére vonatkozó információt nem használt fel az osztályozás elvégzéséhez.

Ez a hiányosság azonban egyszerűen orvosolható néhány extra komponens bevezetésével, amit a deformálható részmodellek[45] meg is tesznek. E módszerek lényege, hogy az egyes osztályokat nem egyetlen vizuális szavakat tartalmazó halmazként, hanem több, különálló szóhalmazból álló struktúraként írják le. Az ilyen modellek esetén az objektumok több részből állnak, amely részek egyesével megfeleltethetők egy-egy vizuális szavakból álló halmazosztálynak, azonban e részek együttesen alkotnak egyetlen szemantikus osztályt.



26. ábra

A deformálható részmodell

Forrás: a szerző felvétele és szerkesztése

[45] Felzenszwalb, Pedro – Girshick, Ross – McAllester, David – Ramanan, Deva (2013): Visual Object Detection with Deformable Part Models. *Communications of the ACM,* Vol. 56, No. 9. 97–105.

A modell e részeknek a központtól számított relatív pozícióját és azok szórását minden egyes részhez külön megtanulja. Szükséges azonban még az is, hogy a modell megtanulja azt is, hogy mely vizuális szavak melyik részhez tartoznak az osztályon belül, ugyanis a szótár konstruálásához használt adatbázisban csak arra vonatkozó információ van, hogy az egyes szavak melyik osztályhoz tartoznak, ezen belül külön információval a részekről nem rendelkezünk.

Fontos megjegyezni, hogy ahogyan az eredeti vizuális szóhalmazeljárás igényelte, hogy a szótár konstruálásához használt adatbázisban a képek legyenek felcímkézve az azon megtalálható osztályok szerint, a deformálható részmodelleknek ezenfelül szüksége van az objektumok pozícióinformációjára is. Ilyen adatbázisokat első sorban kézzel szoktunk előállítani, majd ezután az így elkészült algoritmus képes lesz arra, hogy más, eddig ismeretlen képekre is az adott műveletet elvégezze.

Ha az objektumdetektálást absztrakt osztályok szintjén kívánjuk elvégezni, akkor gyakori eljárás, hogy a feladat megoldásához egy objektumjavasló és egy osztályozó módszer együttesét használjuk. Az objektumjavasló módszer feladata, hogy a képen olyan részleteket találjon, amelyek nagy valószínűséggel tartalmaznak valamilyen egybefüggő objektumot. Ezt követően az összes ilyen javaslatnak megfelelő részletet külön-külön kivágjuk a képből, és egy osztályozó algoritmus segítségével megkapjuk a címkéjüket. Természetesen a fenti eljárást meg lehetne valósítani javasló módszer használata nélkül is egy mozgóablakos séma használatával, ez azonban a végigpróbálandó pozíciók és skálák száma miatt rendkívül költséges volna.

Az objektumjavasló módszereket alapvetően két részre oszthatjuk: az egyik a régióalapú javaslás, míg a másik pedig az ablakalapú módszer. Az előbbi típusba tartozó módszerek valamilyen módon egy hasonló tulajdonságú pixelekből álló összefüggő régiót igyekeznek találni a képen, és ezekből a régiókból készítenek objektumjelöltet. Ez alapján voltaképpen ezek képszegmentáló eljárások, amelyeket részletesen a következő alfejezet tárgyal.

Az ablakmódszerek ezzel ellentétben nagyszámú téglalap alakú ablakot javasolnak véletlen generálás vagy valamilyen séma alapján (leggyakrabban a kettő vegyítésével). Ezekre az ablakokra pedig egyesével kiértékelnek egy objektumszerűség-értéket, amely annak a valószínűsége, hogy az adott ablak magában foglal egy objektumot. Ezt az értéket számos jellemző felhasználásával, heurisztikus módszerekkel lehet számolni. Példaképpen felhasználható az, hogy azok az ablakok, amelyek lényeges több zárt kontúrt

tartalmaznak teljesen, mint amennyit elmetszenek, nagyobb valószínűséggel tartalmaznak egy objektumot.[46]

Az objektumdetektálás és az osztályozás területén is elterjedtek továbbá a rendkívül népszerű, mély neurális hálókat alkalmazó módszerek, amelyek a mélytanulás témakörébe tartoznak. Ezeket a kismonográfia második kötete fogja részletesen ismertetni.

Alkalmazás: Ahogy azt az előző alfejezet végén is említettük, a különböző megfigyelőfunkciót ellátó látórendszerek esetén fontos lehet a látott objektumok osztályozása, hiszen ettől függően más döntések és akciók elvégzése lehet célszerű. Ezeket a döntéseket és akciókat viszont ugyanúgy befolyásolhatják a detektálás műveletéből kinyert információk, legfőképp a detektált kategóriák pozíciója és száma. Például forgalommonitorozás esetén az autók számából következtethetünk a forgalom állapotára, valamint ember észlelése esetén megkülönböztethetjük azt a normális esetet, amikor az ember a leállósávban található, attól az azonnali riasztást igénylőtől, akit valamelyik forgalmi sávban detektáltuk.[47]

4.3. Szegmentálás

A számítógépes látás megoldásai során sok esetben van arra szükségünk, hogy a kép pixeleit egyesével különálló objektumokhoz soroljuk, vagyis a képet szegmentáljuk. A szegmentálás kimenete leggyakrabban egy két- vagy többállapotú kép, amely adott esetben akár maszkként is felhasználható arra, hogy az eredeti képen található objektumot a képből kiemeljük vagy kivágjuk. A szegmentálás művelete felhasználható az előző alfejezet végén említett objektumjavasló módszerek létrehozására is.

A felhasznált információk és az eljárás konkrét kimenete alapján a szegmentálás többféle változatát különböztethetjük meg. Előtér-szegmentálásról beszélhetünk, ha valamilyen ismert tulajdonságú objektumhoz tartozó pixeleket kívánunk megkeresni. Hagyományos képszegmentálás esetén viszont nincs kitüntetett előtérobjektum, hanem a képet egyszerűen különálló objektumok szerint kívánjuk szétválasztani. Előfordul, hogy a képet

---

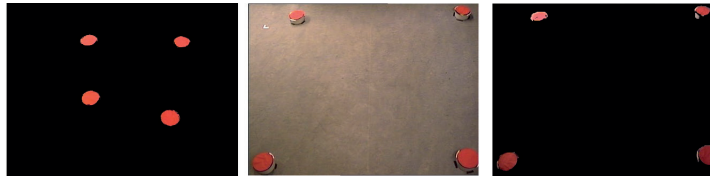
46 Zitnick, C. Lawrence – Dollár, Piotr (2014): *Edge Boxes. Locating Object Proposals from Edges.* Zurich, European Conference on Computer Vision.

47 Varma–Sreeraj 2013.

nem különálló objektumok, hanem objektumkategóriák, -osztályok szerint szegmentáljuk, ekkor szemantikus szegmentálásról beszélünk. Ez a mód azonban ugyanazon osztály két szomszédos objektumát egyetlen szegmenssé olvasztja össze. Ammenyiben nemcsak osztályok, hanem azon belül külön példányok szerint is szeretnénk szegmentálni, akkor példányszegmentálásról beszélünk.

Előtér-szegmentálást legegyszerűbb szín vagy intenzitás alapján végezni. Egyszerű, kontrollált környezetekben gyakran garantálható, hogy az alkalmazásunk számára releváns objektum lesz az egyetlen meghatározott színű tárgy a képen, így egyszerűen egy, a színcsatornákon elvégzett küszöbözés segítségével a szegmentálás elvégezhető. Természetesen a valóságban az így kapott bináris képet még a 4.4. alfejezetben ismertetett módszerek segítségével szűrni kell. Fontos megjegyezni, hogy akár intenzitás-, akár színalapú szegmentálásról beszélünk, a küszöbözést szinte kizárólag a HSV- vagy a YCbCr-színterek valamelyikében (vagy ezek variációiban) végzik el.

A küszöbözés alapú szegmentáció esetén gyakran a tervezők határozzák meg a használt küszöbértéket, amely gyakran szuboptimális lehet. Ennek elkerülésére gyakorta alkalmazzuk a hisztogram-visszavetítés módszerét.[48] Az eljárás elején a szegmentálandó objektumról egy referenciahisztogramot készítünk, amelyet aztán a működés során a képpel összehasonlítunk. A képen a visszavetítés során minden pixelhez meghatározzuk, hogy mekkora eséllyel származik a referenciahisztogramból. Az így kapott valószínűségi képet küszöbözve kaphatunk egy maszkot, amely az objektumhoz tartozó pixeleket tartalmazza.



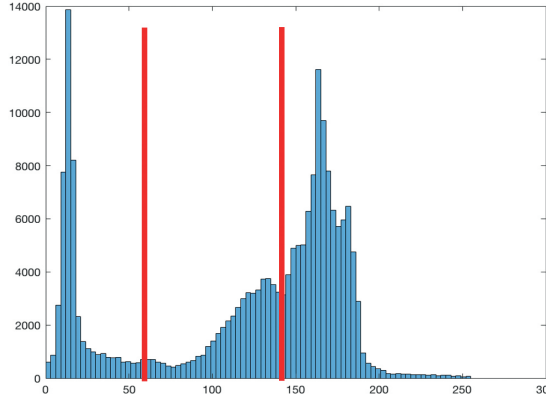
27. ábra

Referenciakép (bal), szegmentálandó kép (közép), hátravetítés segítségével szegmentált kép (jobb)

Forrás: a szerző szerkesztése

[48] Russ 2011.

A szegmentálandó kép hisztogramját felhasználhatjuk hagyományos szegmentáláshoz is, referenciaobjektum nélkül. Ekkor az algoritmus a kép hisztogramjában völgyekkel elválasztott csúcsokat keres, ily módon több részre osztva a kép pixeleit. Ezt a szegmentálást természetesen több dimenzióban egyszerre is el lehet végezni, amely esetben mind a szín-, mind a világosságinformációt felhasználhatjuk.



28. ábra

Hisztogramalapú szegmentáció, ahol a piros vonalak jelképezik az automatikusan választott küszöbértékeket

Forrás: a szerző szerkesztése

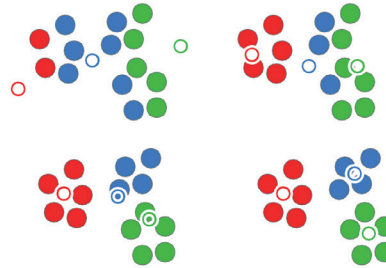
Alkalmazás: Számos virtuális- és kiterjesztettvalóság-rendszer használ kézi gesztusokat az ember-gép kommunikáció egyik irányának megvalósítására.[49] Bár léteznek olyan megoldások, ahol különféle érzékelőkkel ellátott kesztyűk segítségével érzékeljük ezeket a gesztusokat, legalább ugyanannyi kameraalapú gesztusfelismerő rendszer létezik. Ezek működéséhez azonban szükséges a kéz szegmentálása, amelyet könnyű például a hisztogram-visszavetítéses módszer alapján, a bőr színének segítségével elvégezni. Fontos megjegyezni, hogy ezt a megoldást csak akkor célszerű alkalmazni, ha a kézen kívül a kamera nem lát más bőrfelületet, mivel

[49] Sagayam, K. Martin – Hemanth, D. Jude (2017): Hand Posture and Gesture Recognition Techniques for Virtual Reality Applications. A Survey. *Virtual Reality,* Vol. 21, No. 2. 91–107.

ebben az esetben azt is kéznek tekinthetjük. Ez általában garantálható fejre szerelt látórendszerek (például kamerával rendelkező megjelenítő szemüvegek) esetében.

A fenti, hisztogramalapú szegmentáláshoz rendkívül hasonló megoldás a klaszterezésre épülő szegmentálás. Korábban, a vizuális szóhalmazos osztályozás során már említettük a klaszterezést, amelynek lényege, hogy egy tetszőleges térben értelmezett ponthalmaz pontjait úgy osztjuk be valahány részhalmazba, vagyis klaszterbe, hogy az így kapott klaszterek valamilyen kompaktsági kritériumot minél inkább kielégítsenek. Bár klaszterezésre számos algoritmust javasoltak, az egyik legelterjedtebb megoldás a k-közép-eljárás (angolul: k-Means).[50]

A k-közép-módszer a ponthalmazt úgy kívánja k darab klaszterbe sorolni, hogy e klaszterek elemeinek a klaszter középpontjától számított négyzetes távolságainak összege minimális legyen. Ehhez egy iteratív algoritmust használ, amelynek inicializáló lépése, hogy az adatpontok által kifeszített térbe véletlenszerűen lerak k darab középpontot. Ezt követően az algoritmuskonvergenciáig ismételi a következő két lépést. Első lépésként minden pontot hozzárendel a hozzá legközelebb lévő klaszterközépponthoz, így az egyes pontok klaszter-hozzárendelését megváltoztatja. Ezt követően az egyes klaszterközéppontoknak új értéket ad, amely az adott klaszterhez tartozó pontok számtani közepe lesz. Ez a lépés megváltoztatja a középpontok helyzetét, így azt is, hogy melyik pont melyik klaszterhez tartozik, így az iteráció tovább futtatható.



29. ábra

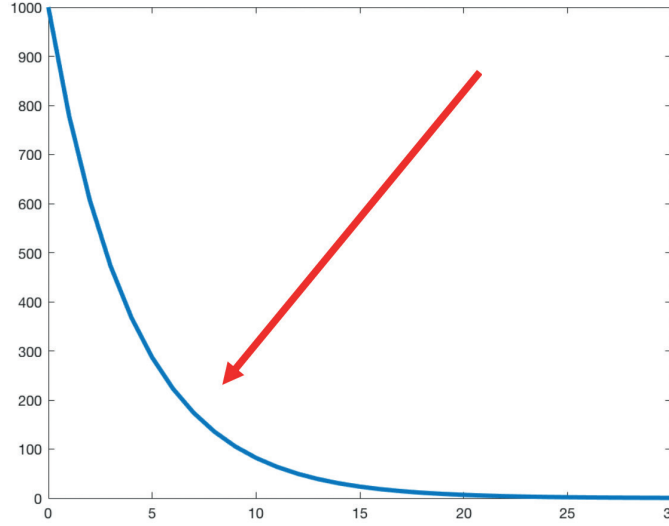
A k-közép-eljárás konvergenciája

** Megjegyzés: az iterációk bal felülről haladnak.*

Forrás: a szerző szerkesztése

[50] Forgy 1965.

A k-közép-módszer bizonyítottan konvergál a véletlen inicializálás konkrét értékeitől függetlenül. Leállási feltételnek választható a klaszterközéppontok vagy a ponthozzárendelések állandósága (vagy adott esetben, ha a változás egy küszöbérték alatt van). Fontos megjegyezni, hogy a módszer nevében és működésében szereplő k egy, a tervező által választott érték, amelynek körültekintő megválasztása nagymértékben befolyásolja az algoritmus működését. A probléma ugyanis az, hogy a k növelésével minden esetben lehetséges a kapott klaszterek kompaktságát növelni, és valóban, N darab pont beosztható N darab klaszterbe zérus hiba mellett. Ez persze azt jelentené, hogy egy kép összes pixelét külön objektumhoz soroljuk, amelynek csekély értelme van. Érdemes ezért megjeleníteni a hiba csökkenését a klaszterek számának függvényében, és kiválasztani ennek a függvénynek a könyökpontját, vagyis azt a klaszterszámot, amelyet növelve a hiba már nem csökken tovább számottevően.



30. ábra
A könyökpont

Forrás: a szerző szerkesztése

Az eddig felsorolt módszereknek az egyik legnagyobb hátránya, hogy csupán az intenzitás- vagy színinformációt veszik figyelembe. Mivel az univerzum legtöbb objektuma térben összefüggő (ebből következően a képük is összefüggő lesz), ezért alapvető kritérium, hogy a szegmentálás során kapott szegmensek is ilyenek legyenek. Erre a problémára adnak megoldást a különböző régióalapú módszerek, például a régiónövesztés és a régiószeletelés módszere, valamint a gráfvágásokon alapuló szegmentálási módszerek.

A régiónövesztés[51] eljárása esetén az algoritmus első lépésként kiválaszt a képről néhány magpontot. Ezeknek a régiómagoknak a választása történhet a képpontok intenzitása alapján vagy például egy rácshálón egyenletesen elhelyezkedve. A régiómagok lesznek az egyes régiók kezdeti értékei. Ezekből a magokból kiindulva megvizsgáljuk minden régió még címkézetlen szomszédjait, és kiértékelünk egy régiótagsági függvényt. Amennyiben ez a tagsági függvény pozitív eredményt ad vissza, akkor a vizsgált pontot hozzáadjuk a régióhoz, ellenkező esetben nem. Az algoritmus leáll, ha már egyetlen pontot sem tudunk egyik régióhoz sem hozzáadni.

A régiónövesztéses algoritmus eredményét számos tényező befolyásolja. Ezek közül az egyik a kiindulópontok megválasztása, amelyeket célszerű a kép hisztogramjának felhasználásával kiválasztani. Érdemes továbbá az eredményben kapott, esetlegesen túl kicsi régiókat elvetni, mert ezek nagy valószínűséggel valamilyen lokális zaj vagy hiba termékei. A legfontosabb tényező azonban a régióhoz tartozás kritériuma. Legyakrabban ezt a szomszédos pixelek közötti intenzitáskülönbség vagy a régi középpont és a vizsgált pixel közötti intenzitáskülönbség alapján döntjük el, egy küszöbérték alapján. Speciális képek esetén fel lehet még használni a szín vagy textúrabeli hasonlóságokat.

Habár a régiónövesztéses módszer egy egyszerű és jól használható módszer, amely ráadásul a kiindulópontok és a kritérium(ok) szabad megválasztása miatt meglehetősen flexibilis, hátránya azonban, hogy meglehetősen számításigényes, valamint a pixelek egyesével történő vizsgálata miatt igencsak érzékeny a zajra, valamint a szomszédosság választására. Ezenfelül, mivel lokális módszer, nem veszi figyelembe az egész képen látható, globális információkat, amelyek a szegmentálást megváltoztatnák.

[51] Russ 2011.

Ezekre a problémákra talál megoldást a régiószeleteléses[52] eljárás, amely kiindulási állapotában az egész képet egyetlen összefüggő szegmensnek tekinti. Ezt követően iteratív módon végighalad az összes szegmensen, és ha a szegmens megfelel egy, a tervező által megadott homogenitási kritériumnak, akkor érintetlenül hagyja. Ellenkező esetben a szegmenst négy, egyenlő területű részre osztja, majd ezeket a szegmenseket is megvizsgálja. Ezt addig folytatja, amíg még keletkeznek új szegmensek.

31. ábra
A régiószeleteléses eljárás elve
Forrás: a szerző szerkesztése

A szeletelés után viszont jó eséllyel szétválasztunk számos, egyébként egybetartozó régiót is. Éppen ezért a szeletelés befejezése után egy összeolvasztási fázis is következik, amely szintén egy hasonlósági kritérium alapján összevon szomszédos régiókat, amennyiben ez szükséges. E módszer használatával sikerült egy gyors, globális információt használó eljárást alkotni, amely lényegesen robusztusabb a zajokra, és majdnem teljesen invariáns a szomszédsági választásra. A módszer hátránya, hogy a szegmensek határa nem mindig lesz teljesen pontos a négyzetes módon végzett szeletelés miatt.

[52] Russ 2011.

Érdemes megemlíteni még a gráfvágásra[53] alapuló szegmentálási módszereket, amelyeknek számos változata létezik. E módszerek közös tulajdonsága, hogy a képet egy súlyozott, irányítatlan gráfként írják le, ahol a gráf csomópontjai a pixelek vagy a lokális pixelcsoportok. A gráf élei csak a szomszédos pixelek vagy csoportok között vannak megadva, a rajtuk szereplő súlyok pedig a pixelek vagy csoportok közötti különbözőséget fejezik ki. A legtöbb ilyen eljárás lényege, hogy a konstruált gráfot olyan módon vágják több részre, hogy e vágások költsége minimális legyen. Ezt természetesen a gráfból kitörölt éleknek a súlyai alapján határozzák meg.

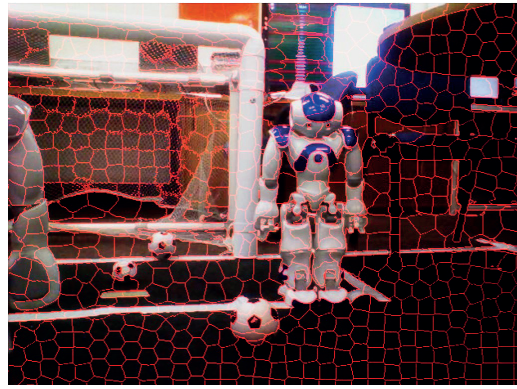
Alkalmazás: Érdemes megjegyezni, hogy szegmentálási módszereket gyakran alkalmaznak úgynevezett szuperpixel-szegmentációra is. A szuperpixel[54] egy relatíve kisméretű, kompakt képrészlet, amely valamilyen homogenitási tulajdonságokkal rendelkezik. Nagyméretű, összetett objektumok általában számos szuperpixelből állnak. A szegmentáció ilyen módon történő elvégzésére ugyanezek az algoritmusok használhatók, csupán a paramétereiket kell úgy beállítani, hogy sok, kisméretű szegmenst találjanak. Léteznek természetesen kifejezetten erre kifejlesztett eljárások. A szuperpixelek rendkívül hasznosak számos alkalmazásban, például a képek kézzel történő felcímkézését (amely szükséges a gépi tanuláshoz való, tanító adatbázisok előállításához) lehet gyorsítani ezzel a módszerrel.

Az alfejezet bevezető részében említést tettünk két másik szegmentálás típusról: a szemantikus, illetve a példányszegmentálásról. Ezeket a feladatokat a modern számítógépes látás gyakorlatában szinte kizárólag mély neurális hálók segítségével végzik, amelyeket a jelen kismonográfia második kötete fog részletesen ismertetni.

---

53 Wu, Z. – Leahy, R. (1993): An Optimal Graph Theoretic Approach to Data Clustering. Theory and Its Application to Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 15, No. 11. 1101–1113.

54 Zhang, Yongxia – Li, Xuemei – Gao, Xifeng – Caiming Zhang (2017): A Simple Algorithm of Superpixel Segmentation With Boundary Constraint. *IEEE Transactions on Circuits and Systems for Video Technology,* Vol. 27, No. 7. 1502–1514.



32. ábra
Szuperpixel-szegmentáció
Forrás: a szerző felvétele és szerkesztése

4.4. Videoanalitika

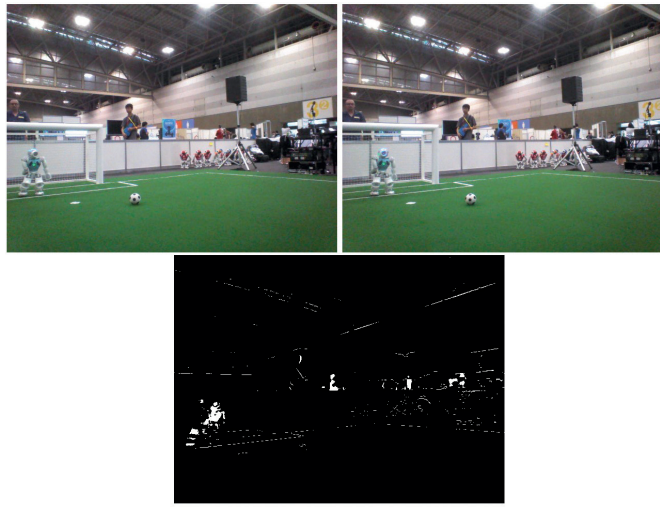
A számítógépes látás tudományterületén belül külön figyelmet szentelünk annak a meglehetősen gyakori esetnek, amikor a feldolgozás alapja nem állókép, hanem egy videó. A videókon végrehajtandó feladatok hasonlók az állóképeken végzendőkhöz. Végezhetünk például szegmentálást, detektálást, valamint osztályozást is, azonban megjelennek újabb feladatok, például a mozgásérzékelés és -követés, valamint a különböző események felismerése.

Ahhoz, hogy ilyen jellegű bemenetek esetén is tudjunk feldolgozást végezni, először tisztázni kell a videók reprezentációjának módját. Az esetek túlnyomó többségében egy videót egyszerűen állóképek sorozataként értelmezünk, amelyeket a beérkezésük sorrendjében dolgozunk fel. Fontos megjegyezni, hogy az eljárásainkat általában úgy alkotjuk meg, hogy nem használjuk fel az éppen feldolgozandó képkockát követő, jövőbeli képkockák által hordozott információt. Ezt esetleg meg lehet tenni, ha nem valósidejű feldolgozás a célunk, ellenkező esetben viszont erre nincs lehetőség. Az állóképek sorozataként történő feldolgozás mellett lehetséges még a videót egy alapkép és az azt követő különbségképek sorozataként is értelmezni, de ezt a számítógépes látás során ritkán alkalmazzuk.

Az első, fontos videoanalitikai feladat a mozgás észlelése és szegmentálása, amely tulajdonképpen az előző alfejezet elején bemutatott előtérszegmentáció egyik formájának is tekinthető, ahol a szegmentálás alapja

nem a szín vagy az intenzitás, hanem a mozgás. A mozgás detektálásának egyik legegyszerűbb módja az eltérő időpillanatban (általában nem rögtön egymás utáni, hanem 0.5–1 másodperc különbséggel) készült különbségkép képzése, amellyel ki lehet emelni azokat a pixeleket, amelyek a képkockák között jelentős változáson mentek keresztül. Az így kapott különbségképet küszöbözzük, így a kapott bináris mozgásképen az „1" pixelek területét kiszámítva dönthetünk arról, hogy érzékeltünk-e mozgást a képen.

A fent leírt, rendkívül egyszerű módszernek számos hátránya van, amelyek közül az egyik, hogy egy mozgó objektum esetén a módszer az objektum mindkét képen lévő pozíciója helyén mozgást fog érzékelni, így a kapott szegmentálásra is használandó mozgáskép „szellemképes" lesz, vagyis egy mozgó objektum két helyen is megtalálható lesz. A másik probléma, hogy számos olyan esemény történhet a videón, amelyet nem tekintünk mozgásnak, mégis jelentős különbséget okoz az egymást követő képkockák intenzitásértékeiben. Erre nagyszerű példa a megvilágítás megváltozása, amely a szabad téren készült videók esetében az egyik legfontosabb zavaró tényező. Beltéri videók esetén is okozhat ilyen változást például a kamera automatikus fehéregyensúly-állító funkciója.



33. ábra

A fenti képek különbsége, ahol számos mozgó objektum kétszer is látható

Forrás: a szerző felvétele és szerkesztése

A fenti problémák kiküszöbölésére való a Gauss-háttérmodellen[55] alapuló mozgásdetektálás. Ennek az eljárásnak a célja, hogy egy statikusnak tekintett hátteret a mozgó előtértől képes legyen elválasztani. Ehhez a háttérről egy statisztikai modellt készít egy mozgóátlagoló eljárás segítségével, és minden egyes pixelhez kiszámítja a pixel értékének várható értékét és szórását. Ezt követően a működés során a különbségképet az aktuális kép és a háttérkép között végezzük, és ezt küszöbözzük. Az így elért eredmények már nem tartalmaznak szellemképet, valamint a fokozatosan végbemenő intenzitásváltozások (például, hogy kisüt a nap) be tudnak épülni a háttérbe hamis mozgás detektálása nélkül.

A háttérmodellben minden pixelhez kiszámolt szórásértéket fel lehet használni az adaptív küszöbérték meghatározására. Ily módon a modell automatikusan lehet érzéketlenebb olyan területeken, ahol gyakori a mozgás, és érzékenyebb ott, ahol ez ritkább. Ez a megoldás olyan esetekben lehet hasznos, ha sok olyan terület látszik a képen, ahol gyakori, irreleváns mozgást érzékelhetünk (például szeles időben a bokrok, illetve fák képe).

Alkalmazás: Térmegfigyelési és biztonsági alkalmazások esetén fontos az elkészült felvételek tárolása a későbbi felhasználás érdekében. Ez általában rendkívüli méretű tárhelyet igényel, nagy részben a videók mérete és a megkövetelt redundancia miatt. A szükséges tárhely azonban nagymértékben csökkenthető, ha csak azokat a felvételrészleteket tároljuk el, ahol mozgást érzékeltünk, mivel azok a felvételek, ahol semmi sem történt, nyilvánvalóan nem értékesek számunkra. E rendszerek működését nagymértékben javítja a zajokra robusztusabb háttérmodelles eljárás alkalmazása, különösképpen kültéri felvételek esetén.

A mozgás felismerése és szegmentálása mellett gyakori feladat lehet annak nagyságát és irányát is meghatározni. Erre egy széles körben elterjedt megoldás az optikaiáramlás-algoritmus.[56] Ennek alapelve, hogy a képet minden pixelben lokálisan egy lineáris síkfelülettel közelíti, majd a síkfelület meredekségéből és a két kép között történt intenzitásváltozás értékéből következtet a mozgás mértékére. Az optikai áramlás feltételezi, hogy képen az objektumok intenzitása nem változik a két kép között, csupán

---

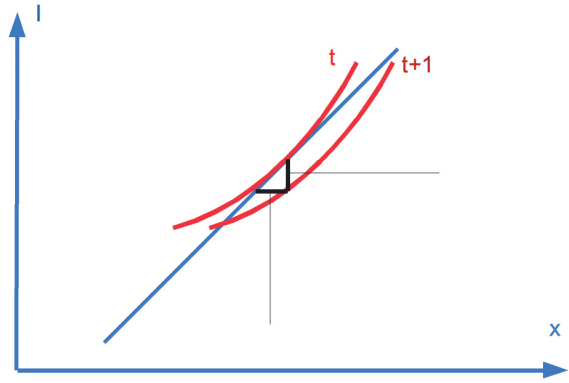
55 Wren, Christopher – Azarbayejani, Ali J. – Darrel, Trevor J. – Pentland, Alexander P. (1997): Pfinder: Real-Time Tracking of the Human Body. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 19, No. 7. 780–785.

56 Kató–Czúni 2001.

elmozdulás történik. A mozgás becslésére az optikaáramlás-algoritmus az alábbi intenzitásáramlás-egyenletet használja:

$$\frac{I_x}{I_t}u + \frac{I_y}{I_t}v = -1$$

Ahol I_x,I_y és I_t a képirányok és az idő szerinti deriváltjai, u és v pedig a mozgás sebessége az egyes irányokban. A kép deriváltjait numerikusan szűrők és különbségképzés segítségével tudjuk számolni, azonban így is egyetlen egyenlet jut két ismeretlenre, amiből következik, hogy az intenzitásáramlás-egyenlet nem oldható meg egyértelműen. A valóságban az egyszerű áramlásmódszer csupán a mozgásvektornak az egyik – a képi gradiens irányába eső – komponensét képes meghatározni.



34. ábra

Az optikai áramlás alapelve egy dimenzióban

Forrás: a szerző szerkesztése

A gyakorlatban erre a problémára az egyik széles körben használt megoldás a Lucas–Kanade-féle optikaiáramlás-algoritmus. Ez az eljárás eggyel több feltételezést eszközöl, mégpedig azt, hogy a képen egymáshoz közel található pixelek nagy valószínűséggel együtt is mozognak. Ennek alapján a Lucas–Kanade-módszer[57] nemcsak az adott pixel pozíciójában kívánja megoldani

[57] Lucas, B. D. – Kanade, Takeo (1981): *An Iterative Image Registration Technique with an Application to Stereo Vision.* Proceedings of Imaging Understanding Workshop.

az intenzitásáramlás-egyenletet, hanem annak egy bizonyos környezetében. Erre a legkisebb négyzetek módszerét alkalmazza az alábbi módon:

$$\begin{bmatrix} I_x^1 & I_y^1 \\ I_x^2 & I_y^2 \\ \vdots & \vdots \\ I_x^N & I_y^N \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t^1 \\ I_t^2 \\ \vdots \\ I_t^N \end{bmatrix} \Rightarrow Xu = Y \xrightarrow{LS\ becslés} u = (X^T X)^{-1} X^T Y$$

Fontos megjegyezni, hogy az egyenletben szereplő X^TX mátrix a korábban ismertetett KLT-sarokdetektorban is használt lokálisstruktúra-mátrix (ez nem meglepő, hiszen a KLT-detektor nevében szereplő Kanade és Lucas ugyanazok a kutatók, akikről a jelenleg tárgyalt módszer is az elnevezését kapta). Ha visszaemlékezünk a lokálisstruktúra-mátrixról folytatott diszkusszióra, tudhatjuk, hogy ennek a mátrixnak a sajátértékei a kép legkisebb és legnagyobb változásának nagyságát fejezik ki. Ahhoz, hogy ezt a mátrixot invertálni tudjuk, mindkét sajátértéknek jelentősen különbözni kell a nullától, vagyis a Lucas–Kanade-módszert csak sarokszerű pontok esetén lehet kiértékelni. Éppen ezért ez a módszer ritka optikaiáramlás-módszerként is ismert, mivel csak a kép néhány pontján számolható ki.

Létezik azonban sűrű optikaiáramlás-algoritmus is, amely képes egyértelműen meghatározni az áramlás irányát, és minden pontban számítható is. Ez az eljárás Gunnar Farneback[58] nevéhez fűződik. Ennek alapvető lényege, hogy a képet lokálisan nem egy lineáris felülettel (síkkal), hanem egy másodfokú polinommal közelíti. Az eredeti áramlásalgoritmushoz hasonlóan feltételezi, hogy a két képkocka között nem történik intenzitásváltozás, csupán elmozdulás. Innen a polinomokat mindkét képkockán kiszámolva és összevetve az egyes pixelek elmozdulása meghatározható.

Alkalmazás: Az optikai áramlást számos területen szokás alkalmazni, amelyek közül az egyik jelentős a mozgókamerás 3D-rekonstrukciók végzése.[59] Az algoritmus használatával lehetőség nyílik arra, hogy egy arra egyébként nem alkalmas, egyszerű kamera segítségével térbeli felvételeket készítsünk. Ehhez a kamerával egy, az objektumot körbejáró videót készítünk,

---

58 Gunnar, Farnebäck (2003): *Two-Frame Motion Estimation Based on Polynomial Expansion.* Scandinavian Conference on Image Analysis, Halmstad.

59 Diskin, Yakov – Asari, Vijayan K. (2013): *Dense 3D Point-Cloud Model Using Optical Flow for a Monocular Reconstruction System.* Washington, IEEE Applied Imagery Pattern Recognition Workshop.

majd az optikai áramlás segítségével az egyes pixelek elmozdulását kiszámoljuk. Az elmozdulás mértékéből a kamerától való távolságra tudunk következtetni (a távoli objektumról származó pixelek kisebb sebességgel mozognak a képen).

További érdekes felhasználás a különböző mozgásesemények osztályozása. A képen mozgó objektumokat ugyanis az áramlás képe alapján beoszthatjuk különböző irányokba haladó, forgó, illetve kamerához közeledő vagy attól távolodó objektumokba. Ez rendkívül hasznos, ha szeretnénk észlelni vagy elkerülni a kamerának helyet adó berendezéshez való közeledést vagy az azzal való ütközést.

Az objektumok követését elvégezhetjük még természetesen gradienshisztogram-alapú lokális képjellemzők segítségével is.[60] Ennek módja szinte teljesen megegyezik az objektumdetektálás módszerével. Az egyetlen lényegi különbség, hogy – feltételezve azt, hogy a követett objektum mozgási sebessége korlátos – nincs szükség a képjellemzők egész képen történő keresésére, elég csupán a korábbi detektálás közelében keresni a képjellemzőket, amely az egyébként meglehetősen lassú procedúrát lényegesen képes gyorsítani. Ezenfelül, mivel a párosítást is csak közeli képjellemzők között kell elvégezni, ez is gyorsabb és robusztusabb lesz.

Fontos megjegyezni azonban, hogy ha a követésre mindig az előző képkockán észlelt képjellemzőket használjuk fel, akkor előfordulhat, hogy a követés hibái az egymást követő képkockákon összeadódnak, és a követésben egy egyre akkumulálódó hiba, az úgynevezett driftelés következik be. Ezt a hibát ellensúlyozandó célszerű, ha bizonyos időközönként elvégzünk egy újradetektáló lépést, amely során egy referenciaobjektum alapján az egész képen keressük az objektumot. Mivel a detektálás művelete relatíve drága, ezért – a valósidejű működést biztosítandó – ezt nem lehet minden lépésben elvégezni.

A detektálásban röviden említett módszerek esetén az objektum akár hat szabadságfokú módon is követhető, feltéve, hogy az egyes képjellemzők térbeli pozícióját az objektum felületén ismerjük. Egyéb esetben szükségünk van valamilyen extra háromdimenziós információra, amelyet 3D

---

60 Sakai, Yuki – Oda, Tetsuya – Ikeda, Makoto – Barolli, Leonard (2015): *An Object Tracking System Based on SIFT and SURF Feature Extraction Methods.* Taipei, 18th International Conference on Network-Based Information Systems.

rekonstrukció, vagy egy RGB-D kamera használata segítségével nyerhetünk.[61] A kétdimenziós képek térbeliségének visszaállítását a kismonográfia második kötetében tárgyaljuk részletesen.

Alkalmazás: A videókon történő mozgáskövetésnek számos gyakorlati alkalmazása van. Egyrészt a különböző mozgó objektumok pályáit hosszabb felvételek esetén megjegyezhetjük, csoportosíthatjuk, amelynek segítségével megkaphatjuk a kamera által figyel területen a mozgó objektumok által gyakran használt útvonalakat. Ezeket az útvonalakat aztán fel lehet használni arra, hogy a megszokottól eltérő mozgásokat, anomáliákat detektálhassunk, amelyeket esetleg egy emberi megfigyelő felé tudunk továbbítani.

Hasonló módon hasznos lehet, ha a mozgások alapján különféle eseményeket vagyunk képesek detektálni.[62] Ehhez a mozgás különböző tulajdonságait lehet felhasználni, például a megszokott pályákat, megállásokat vagy a kamera képén a felhasználók által definiálható területeket. Ilyen módszerekkel képesek vagyunk detektálni, ha egy mozgó objektum belép egy bizonyos területre, vagy például ha egy objektum hosszabb ideig áll egy pozícióban. E tudást a detektálás fejezetében leírtakkal kombinálva észlelhetjük, ha például egy csomagot őrizetlenül hagynak a megfigyelt területen.

---

61 Song, Shuran – Xiao, Jianxiong (2013): *Tracking Revisited Using RGBD Camera. Unified Benchmark and Baselines.* Sydney, IEEE International Conference on Computer Vision.

62 Zhao, Zhicheng – Li, Xuanchong – Xingzhong, Du – Chen, Qi – Zhao, Yanyun – Su, Fei – Chang, Xiaojun – Hauptmann, Alexander G. (2018): A Unified Framework with a Benchmark Dataset for Surveillance Event Detection. *Neurocomputing,* Vol. 278, 62–74.

Vákát oldal

Összefoglalás

A jelenlegi írás segítségével az olvasó betekintést nyerhetett a képfeldolgozás és a számítógépes látás legfontosabb alapfeladataiba és az azokra használt legfontosabb algoritmusok működésébe és tulajdonságaiba. Ezenfelül jó néhány szemléletes alkalmazási példát szolgáltattunk, amelyek segítségével ezen egyszerű eljárások gyakorlatban történő alkalmazási lehetőségeire is rávilágítottunk. Az írás alapvető célja volt, hogy az algoritmusok relevanciáját a közigazgatás szempontjából közvetlenül vagy közvetve (például virtuális- és kiterjesztettvalóság-rendszerekben történő használat) releváns alkalmazásokon keresztül demonstrálja. Virtuális- és kiterjesztettvalóság-rendszerek alkalmazásáról bővebben a kismonográfia-sorozat *Tartalommegjelenítés és ember-gép interakció*[63], valamint az *Eljárások és módszerek virtuálisvalóság-rendszerekben*[64] című köteteiben olvashatnak.

A kismonográfia második kötetében az olvasó betekintést nyerhet a számítógépes látás mesterfokának számító komplex algoritmusok működésébe és használatába. A kötet részletesen ismertetni fogja a háromdimenziós látás és feldolgozás, valamint a manapság rendkívüli népszerűségnek örvendő, gépi tanulásra alapuló látás módszereit. Külön részletességgel ismertetjük majd a mélytanulás-tudományterület (deep learning) módszereit és megoldásait, amelyek az utóbbi néhány év slágerterületének számítanak. Ezenfelül a kötet végső részében részletesen fogunk ismertetni több, a közigazgatás szempontjából releváns, komplex látórendszerekre épülő megoldást is.

---

63 Vajda Ferenc – Jakab László (2019): *Tartalommegjelenítés és ember-gép interakció.* Kézirat megjelenés alatt.

64 Umenhoffer Tamás (2019): *Eljárások és módszerek virtuálisvalóság-rendszerekben.* Kézirat megjelenés alatt.

Vákát oldal

Felhasznált irodalom

Altman, Naomi S. (1992): An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician,* Vol. 46, No. 3. 175–185.

Bay, Herbert – Ess, Andreas – Tuytelaars, Tinne – Gool, Luc Van (2008): SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding,* Vol. 110, No. 3. 346–359.

Canny, John (1986): A Computational Approach To Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 8, No. 6. 679–698.

Cooley, James W. – Tukey, John W. (1965): An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation,* Vol. 19, No. 90. 297–301.

Cooray, Thilan – Fernando, Shilan (2011): *Visual-Based Automatic Coin Counting System.* SAITM Research Symposium on Engineering Advancements.

Cracknell, Arthur P. – Hayes, Ladson (2007): *Introduction to Remote Sensing.* London, Taylor and Francis.

Diskin, Yakov – Asari, Vijayan K. (2013): *Dense 3D Point-Cloud Model Using Optical Flow for a Monocular Reconstruction System.* Washington, IEEE Applied Imagery Pattern Recognition Workshop.

Duda, Richard O. – Hart, Peter E. (1972) Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM,* Vol. 15, No. 1. 11–15.

Erteza, Ahmed (1976): Sharpness Index and Its Application to Focus Control. *Applied Optics,* Vol. 15, No. 4. 877–881.

Felzenszwalb, Pedro – Girshick, Ross – McAllester, David – Ramanan, Deva (2013): Visual Object Detection with Deformable Part Models. *Communications of the ACM,* Vol. 56, No. 9. 97–105.

Fernandes, Leonardo A. – Oliveira, Manuel M. (2008): Real-Time Line Detection Through an Improved Hough Transform Voting Scheme. *Pattern Recognition,* Vol. 41, No. 1. 299–314.

Floriani, Leila D. – Spagnuolo, Michela (2007): *Shape Analysis and Structuring.* London, Springer.

Forgy, E. W. (1965): Cluster Analysis of Multivariate Data. Efficiency Versus Interpretability of Classifications. *Biometrics,* Vol. 21, No. 3. 768–769.

Freeman, Herbert (1961): On the Encoding of Arbitrary Geometric Configurations. *IRE Transactions on Electronic Computers,* Vol. 10, No. 2. 260–268.

Ghadarghadar, Nastaran – Ataer-Cansizoglu, Esra – Zhang, Peng – Erdogmus, Deniz (2012): *A SIFT-Point Distribution-Based Method for Head Pose Estimation.* IEEE International Workshop on Machine Learning for Signal Processing, Santander.

Gunnar, Farnebäck (2003): *Two-Frame Motion Estimation Based on Polynomial Expansion.* Scandinavian Conference on Image Analysis, Halmstad.

Harris, Chris – Stephens, Mike (1988): *A Combined Corner and Edge Detector.* Proceedings of the 4th Alvey Vision Conference.

Kató Zoltán – Czúni László (2001): *Számítógépes látás.* Budapest, Typotex.

Kaur, Mandip – Jindal, Simpel (2013): An Integrated Skew Detection and Correction Using Fast Fourier Transform and DCT. *International Journal of Scientific & Technology Research,* Vol. 2, No. 12. 164–169.

Li, Fei-Fei – Perona, Pietro (2005): *A Bayesian Hierarchical Model for Learning Natural Scene Categories.* IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

Lowe, David G. (2004): Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision,* Vol. 60, No. 2. 91–110.

Lucas, B. D. – Kanade, Takeo (1981): *An Iterative Image Registration Technique with an Application to Stereo Vision.* Proceedings of Imaging Understanding Workshop.

Prewitt, J. (1970): Object Enhancement and Extraction. In Linkin, B. S. – Rosenfeld A. eds.: *Picture Processing and Psychopictorics.* New York, Academic Press.

Rublee, Ethan – Rabaud, Vincent – Konolige, Kurt – Bradski, Gary (2011): *ORB: An Efficient Alternative to SIFT or SURF.* IEEE International Conference on Computer Vision.

Russ, John C. (2011): *The Image Processing Handbook.* Boca Raton, CRC Press.

Sagayam, K. Martin – Hemanth, D. Jude (2017): Hand Posture and Gesture Recognition Techniques for Virtual Reality Applications. A Survey. *Virtual Reality,* Vol. 21, No. 2. 91–107.

Sakai, Yuki – Oda, Tetsuya – Ikeda, Makoto – Barolli, Leonard (2015): *An Object Tracking System Based on SIFT and SURF Feature Extraction Methods.* Taipei, 18th International Conference on Network-Based Information Systems.

Schantz, Herbert F. (1982): *The History of OCR, Optical Character Recognition.* Manchester, Recognition Technologies Users Association.

Sobel, Irwin (2014): *Isotropic 3x3 Image Gradient Operator.* Presentation at Stanford A.I. Project 1968.

Song, Shuran – Xiao, Jianxiong (2013): *Tracking Revisited Using RGBD Camera. Unified Benchmark and Baselines.* Sydney, IEEE International Conference on Computer Vision.

Tomasi, Carlo – Kanade, Takeo (2004): Detection and Tracking of Point Features. *Pattern Recognition,* Vol. 37, 165–168.

Uchiyama, Hideaki – Marchand, Eric (2012): *Object Detection and Pose Tracking for Augmented Reality. Recent Approaches.* 18th Korea–Japan Joint Workshop on Frontiers of Computer Vision, Kawasaki.

Umenhoffer Tamás (2019): *Eljárások és módszerek virtuálisvalóság-rendszerekben.* Kézirat megjelenés alatt.

Vajda Ferenc – Jakab László (2019): *Tartalommegjelenítés és ember-gép interakció.* Kézirat megjelenés alatt.

Varma, Soumya – Sreeraj, M. (2013): *Object Detection and Classification in Surveillance System.* IEEE Recent Advances in Intelligent Computational Systems. Trivandrum, India.

Vincent, Luc – Soille, Pierre (1991): Watersheds in Digital Spaces. An Efficient Algorithm Based on Immersion Simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 13, No. 1. 583–598.

Wren, Christopher – Azarbayejani, Ali J. – Darrel, Trevor J. – Pentland, Alexander P. (1997): Pfinder: Real-Time Tracking of the Human Body. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 19, No. 7. 780–785.

Wu, Z. – Leahy, R. (1993): An Optimal Graph Theoretic Approach to Data Clustering. Theory and Its Application to Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 15, No. 11. 1101–1113.

Zhang, Yongxia – Li, Xuemei – Gao, Xifeng – Caiming Zhang (2017): A Simple Algorithm of Superpixel Segmentation With Boundary Constraint. *IEEE Transactions on Circuits and Systems for Video Technology,* Vol. 27, No. 7. 1502–1514.

Zhao, Zhicheng – Li, Xuanchong – Xingzhong, Du – Chen, Qi – Zhao, Yanyun – Su, Fei – Chang, Xiaojun – Hauptmann, Alexander G. (2018): A Unified Framework with a Benchmark Dataset for Surveillance Event Detection. *Neurocomputing,* Vol. 278, 62–74.

Zitnick, C. Lawrence – Dollár, Piotr (2014): *Edge Boxes. Locating Object Proposals from Edges.* Zurich, European Conference on Computer Vision.

A Dialóg Campus Kiadó a Nemzeti Közszolgálati Egyetem könyvkiadója.

Nordex Nonprofit Kft. – Dialóg Campus Kiadó
www.dialogcampus.hu
www.uni-nke.hu
1083 Budapest, Ludovika tér 2.
Telefon: (30) 426 6116
E-mail: kiado@uni-nke.hu

A kiadásért felel: Petró Ildikó ügyvezető
Felelős szerkesztő: Dalloul Zaynab
Olvasószerkesztő: Sós Dóra Gabriella
Korrektor: Szarvas Melinda
Tördelőszerkesztő: Fehér Angéla
Nyomdai kivitelezés: Pátria Nyomda Zrt.
Felelős vezető: Simon László vezérigazgató

ISBN 978-615-5945-36-6 (nyomtatott)
ISBN 978-615-5945-37-3 (elektronikus)
ISSN 2631-1259

A szerző jelen kötetében a mesterségesintelligencia-alapú számítógépes látásra irányuló kutatásait mutatja be, különös tekintettel a tudományterület alapvető feladataira és nehézségeire, illetve az ezek megoldására szolgáló algoritmikus megfejtésekre. Ezen felül a kismonográfia külön kiemeli a bemutatott eredmények gyakorlati közigazgatási felhasználási lehetőségeit. A mű célja, hogy bevezesse az érdeklődő olvasót a számítógépes látás területének alapjaiba, és a megoldások ismertetésén felül segítse őt annak megértésében, hogy milyen típusú feladatok abszolválhatók könnyedén, és milyen szituációk okozhatnak nehézséget ezeknek az algoritmusoknak.

A kiadvány a KÖFOP-2.1.2-VEKOP-15-2016-00001
„A jó kormányzást megalapozó közszolgálat-fejlesztés”
című projekt keretében jelent meg.