Gépjárművezetés biztonságnövelése képfeldolgozás alkalmazásával

Bencsik Blanka

Felkészítő tanár: Kőrösi Gábor
Bolyai Tehetséggondozó Gimnázium, Posta u. 18., Zenta 2440 Szerbia

1. Bevezetés

A világon rengeteg ember veszti életét közúti balesetben az emberi hibáknak betudhatóan. Ez motivált arra, hogy létrehozzak egy programot, mely könnyíti az autóvezető számára a vezetést, s ezáltal csökkentheti a közúti baleset bekövetkezésének valószínűségét. Az elkészült program képes megállapítani a gépkocsira rögzített kamera által készített felvételből, hogy az úton hol vannak az sávjelző vonalak, hol halad az autó, helyesen tartja-e a sávot, merre kanyarodik az út, és, hogy szaggatott-e a vonal. Mindezekről a vezető grafikus formában kap értesítést, valamint, ha túlzottan kitér a sávból, vagy telt vonalnál előzésbe kezd, akkor figyelmeztető hangjelzést ad. A program elkészítéséhez a Microsoft Visual Studio 2010 szoftvercsomagot, valamint az OpenCV képfeldolgozási könyvtárat használtam.

2. Probléma megoldásának menete

A program a videót képkockákra bontja, s minden képen elvégzi a programkódba foglalt műveleteket.

2.1. Előfeldolgozás

Előfeldolgozás során javítjuk a kép minőségét, előkészítjük, alkalmassá tesszük a feldolgozásra. A színek ebben az esetben nem fognak különösebb szerepet játszani, ezért a képet szürkévé alakítjuk.

2.2. Szegmentálás

A szegmentálás az értékes részek elválasztása a háttértől. Jelen esetben adaptív küszöbölést alkalmazunk (adaptive thresholding), amely nem az egész képre, hanem külön-külön annak kisebb tartományaira keres küszöbértéket. Adaptív küszöböléskor az adott tartományba eső képpontok közül megkeressük a legkisebbet és a legnagyobbat. A két érték különbségének meg kell haladnia egy meghatározott differenciaértéket, különben az eltérés csupán szennyeződésnek tudható be. Ekkor az adott tartományba eső képpontok mindegyike 0 értéket kap (fekete). Ha lokális tartományon belüli maximum és minimum különbsége eleget is tesz a feltételnek, egy-egy képpont csak akkor nyilvánítható fehérnek, ha értéke közelebb van a maximumhoz, mint a minimumhoz. [6] [7] [8]

A küszöbölést csak a kép alsó felén végezzük el, feltételezve, hogy a kép felső felében égbolt, azaz jelentéktelen információ van, azt egyszerűen lefeketítjük.

2.3. Élkiemelés

Sávfelismeréskor a sávjelölő vonalak belső széléhez illesztett vonalat keressük. Vonalfelismerést csak éleket tartalmazó képen lehet eredményesen véghezvinni, ezért a képen élkiemelést (edge detection) kell végezni.

Digitális képen élek ott találhatóak, ahol a szomszédos pixelek között nagy az intenzitáskülönbség. A gradiens számításon alapuló élkiemelési módszerek közös jellemzője, hogy két, egymásra merőleges irányban számol gradienst, mindkettőnek megvan a saját operátora. [1] [3][6] [7] [8] Gradiens:

$$\nabla f = \frac{\partial f}{\partial x}\vec{\imath} + \frac{\partial f}{\partial y}\vec{\jmath} \qquad [2]$$

, ahol $\vec{\imath}$ és $\vec{\jmath}$ az egységvektorok, x és y pedig az irányok.

A két egymásra merőleges irányú differenciából a következőképp határozható meg a gradiens nagysága és iránya:

$$G = \sqrt{{G_x}^2 + {G_y}^2}\ , \quad \varphi = arctg\frac{G_x}{G_y} \qquad [3]$$

2.4. Sávtartás

Bármiféle következtetés levonása előtt ki kell nyerni a képből azt az információt, hogy hol halad az autó, és a meghatározott útsávot grafikusan kijelölni a képen. Az első meghatározásához a sávjelző vonalakhoz illesztett egyeneseket keressük az élkiemelt képen. A vonalfelismeréshez Hough-transzformációt használunk, amely adott kép objektumainak olyan részhalmazát határozza meg, amelyekre közös egyenes illeszthető. [4] Polárkoordináta-rendszerben dolgozunk, mely a sík minden pontját egy szög és egy távolság adattal látja el. Egy egyenes egyenlete így írható le polárkoordináta-rendszerben:

$$y = \left(-\frac{cos\theta}{sin\theta}\right)x + \left(\frac{r}{sin\theta}\right) \quad \Rightarrow \quad r = xcos\theta + ysin\theta \qquad [5]$$

, ahol r a Descartes-féle koordináta-rendszerben az egyenesre az origóból húzott merőleges, θ pedig r és az x-tengely között bezárt szög.

A várt sávjelző vonalak dőlési szögét bizonyos határok közé tudjuk szorítani a kamera állásának alapján, így a több felismert vonal közül ki tudjuk választani azt a kettőt, ami valóban a sávjelző vonalakat jelöli. A vonalnak polárkoordinátában használatos paramétereit ismerjük, x és y koordinátájának meghatározásához analitikus geometriai számításokat kell végeznünk.

Az applikáció egyik szolgáltatása, hogy megállapítja, a gépjármű helyesen halad-e az útsávban. A kamera úgy van felszerelve a kocsira, hogy a sávtartás akkor tökéletes, ha a kép függőleges középvonala egybeesik az útsáv közepével. A program az ettől való százalékos eltérést számolja, a következőképp: $\frac{\Delta x}{h} * 100$, ahol Δx a kép függőleges középvonala és a beazonosított halasási sáv közepe közti differencia, h pedig a beazonosított halasási sáv szélességének fele. Ha a kitérés meghaladja a 20 %-ot a program figyelmeztető jelzést ad a vezetőnek.

2.5. Kanyar felismerése

A kanyar megállapítása perspektivikus képen nehézkes, ezért a képet felülnézetivé transzformáljuk. Az áttranszformált képen az út jobbra kanyarodásakor a sávjelző vonalak jobb irányba, balra kanyarodásakor bal irányba dőlnek, egyenes haladása esetén pedig a két oldalsó sávjelző vonal vagy párhuzamos egymással, vagy kissé egymás felé dől. A program egy vágást végez a kép magasságának háromnegyedén, majd oszloponként összeszámolja a fehér pixeleket, külön a kép magasságának felétől a vágásvonalig, és a vágásvonaltól a kép aljáig terjedő tartományokban. Ezután az alsó tartományban meghatározott értékekből kivonja a felső tartomány értékeit. Ekkor a vágásvonal mentén ahol a sávjelző vonal áthalad rajta, egy hullámot kapunk. A hullám negatívba nyúló része a vágásvonal alatti, pozitívba nyúló része pedig a vágásvonal feletti fehér pixeleket mutatja. A sávjelző vonal haladási iránya abból határozható meg, hogy a hullám melyik tartományba nyúló része következik előbb, vagyis a hullám szélsőértékei között a függvény növekszik vagy csökken. Csökkenéskor balra, növekvéskor jobbra kanyarodik. Az út kanyarodási irányát a program 10 képkockánként határozza meg az addig összeszámolt hullámokból.

2.6. Szaggatott vonal felismerése

A sávjelző vonal folytonosságának vizsgálatát a szegmentált képen végezzük az előzőleg Hough-transzfomációval azonosított vonal mentén, 100 pixeles környezetben, a sávjelző vonalak metszéspontjától a kép aljáig. A program összeszámolja, hogy adott magasságszinteken a vonal 100 pixeles környezetében hány fehér képpont található. Ezután meghatározzuk a küszöbértéket, ami azt mutatja, legkevesebb mennyi fehér képpont szükséges ahhoz, hogy azt mondhassuk, adott magasságszinten megjelenik a sávjelző vonal.

$$h = \frac{1}{2 * db} \sum_{k}^{n} f_k$$

ahol *db* a fehér képpontokat tartalmazó magasságszintek száma, *k* a magasságszint, ahol a vizsgálat kezdetét veszi, *n* a magasságszint, ameddig a vizsgálat tart, f_k pedig a fehér képpontok száma adott magasságszinten.

A küszöbértéken felül eső értékek az mutatják, hogy adott magasságszinten megjelenik e sávjelző vonal. Ezek új, 1-es értéket kapnak, a maradék 0-át. Ezután a program eldönti, hogy a vizsgált vonal folytonos-e. Megszámolja, hogy az egyenes mentén hány olyan magasságszint van, ahol nem jelenik meg a sávjelző vonal. Ha ez szakasz nagyobb, mint a teljes vizsgált szakasz 35 %-a, akkor az sávjelző vonal szaggatottnak, ellenkező esetben teltnek minősül. A felhasználó számára az eredmény a kijelzőn válik láthatóvá: szaggatott vonal esetén egy zöld színű, telt vonal esetében pedig egy piros vonal jelenik meg.

3. Probléma megoldásának menete

Az eredmény egy hatékony, látványos program, mely használata jelentősen megkönnyítheti a gépjárművezető dolgát, s akár a baleset bekövetkezését is megelőzheti. Megalapozott képfeldolgozási eljárásokkal és összetett matematikai számításokkal dolgozik, mely leírása 4 oldalban szinte lehetetlen. Munkám során rengeteg tapasztalatot szereztem. Az itt bemutatott applikáció elkészítése rengeteg munkát és energiát ölel magába, és mégis csak egy része a végleges elképzelésemnek. Mint minden más informatikai fejlesztés, ez az applikáció is, hatékonysága ellenére természetesen bővíthető, továbbfejleszthető a mesterséges intelligencia és gépi látás szakterületére irányulva, s ennek végrehajtása jelen is van a jövőbeli terveim közt.

4. Felhasznált irodalom

[1] Czap László: Képfeldolgozás., Miskolci Egyetem.
[2] Wayne Niblack: An Introduction to Digital Image Processing, Prentice-Hall International Ltd, UK, 1986.
[3] www.inf.u-szeged.hu/~gnemeth/kurzusok/kepfel1/7_gyakorlat.pdf
[4] Fazekas Attila, Kormos János: Digitális képfeldolgozás matematikai alapjai., Debreceni Egyetem Informatikai Intézet.
[5] OpenCV 2.4.13.2 documentation. http://docs.opencv.org/2.4/index.html
[6] Henning Bassmann, Philipp W. Besslich: Ad Oculuos Digital Image Processing., International Thomson Publishing Company, Sheffield, 1995.
[7] Theo Pavlidis: Algorithms for Graphics and Image Processing., Computer Science Press, Rockville, 1982.
[8] Anil K. Jain: Fundamentals of Digital Image Processing., Prentice-Hall, Inc. New Jersey, 1989.