UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Katedra za elektroniku

Predmet: Digitalna obrada slike

Izveštaj: 6. domaći zadatak

Rok za predaju: 9.1.2016.

Student:

Ime	Prezime	broj indeksa
Predrag	Kuzmanović	49/2012

Tačka 1: detekcija ivica

1.1. camerman

Na slici 1.1.1 prikazana je ulazna slika iz koje se idvajaju ivice, **camerman.tif**.

Slika 1.1.1. Ulazna slika, camerman.tif

Kao što se može videti, ulazna slika je veoma zašumljena aditivnim Gausovim šumom. Dakle, potrebno je najpre primeniti niskofrekventno filtriranje uz što bolje zadržavanje ivica na slici kako bi se one na kvalitetan način mogle izdvojiti. Dakle, kao rešenje se nameće bilateralni filtar koji je implementiran u okviru trećeg domaćeg zadatka. Rezultat nakon NF filtriranja pomoću funkcije **bilateral_filter** sa pogodno odabranim parametrima **sigma_s** i **sigma_r** prikazan je na slici 1.1.2.

Slika 1.1.2. Slika nakon NF filtriranja bilateralnim filtrom, camerman_bilateral.tif

Ovako dobijena slika je sada mnogo pogodnija za izdvajanje ivica pomoću gradijenta, na primer pomoću Zobelovog operatora. Rezultat koji se dobija nakon primene ovog gradijentnog operatora i nakon toga poređenja sa pragom prikazan je na slici 1.2.3 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici. Pokušano je računanje gradijenta i pomoću drugih operatora. Ostali operatori sa maskom dimnzija 3x3 daju vrlo slične rezultate kao i Zobelov operator, a operatori sa maskom dimenzija 7x7 (boxcar operatori) daju znatno lošiju lokalizaciju ivica (izdvojene ivice su deblje).

Slika 1.1.3. Izdvojene ivice, camerman_edges.tif

1.2. house

Na slici 1.2.1 prikazana je ulazna slika iz koje se idvajaju ivice, **house.tif**.

Slika 1.2.1. Ulazna slika, house.tif

Kao što se može videti, ulazna slika je poprilično čista (nezašumljena) i sadrži mnogo detalja na slici kao što su ciglice i crepovi na kući. S obzirom na to, zaključujemo da je pogodno primeniti "oštrije" izdvajanje ivica, odnosno može se upotrebiti LoG (Laplasijan Gausove funkcije) filtar. Implusni odziv ovog filtra napravljen je pomoću MATLAB funkcije **meshgrid**. Prilikom izbora parametara standardne devijacije Gausove funkcije i dimenzija maske filtra vodilo se računa da se izbegne preklapanje u spektralnom domenu. Rezultat koji se dobija nakon primene LoG filtra prikazan je na slici 1.2.2.

Slika 1.2.2. Slika nakon primene LoG filtra, house_LoG.tif

Nakon ovoga, vrši se binarizacija (poređenje sa pragom) dobijene slike kako bi se detektovale ivice. Rezultat koji se dobija za optimalnu vrednost praga (što veći procenat detekcije pravih ivica i što manji procenat detekcije lažnih ivica) prikazan je na slici 1.2.3 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici.

Slika 1.2.3. Izdvojene ivice, house_edges.tif

1.3. lena

Na slici 1.3.1 prikazana je ulazna slika iz koje se idvajaju ivice, **lena.tif**.

Slika 1.3.1. Ulazna slika, lena.tif

Kao što se može videti, ulazna slika je blago zašumljena aditivnim Gausovim šumom. Dakle, potrebno je najpre primeniti blago niskofrekventno filtriranje kako bi se smanjila detekcija lažnih ivica. Filtriranje je u ovom slučaju odrađeno niskofrekventnim Gausovim filtrom odgovarajuće standarnde devijacije, vodeći računa da se ne izgube detalji na slici. Rezultat nakon NF filtriranja prikazan je na slici 1.3.2.

Slika 1.3.2. Slika nakon NF filtriranja, lena_lp.tif

Ovako dobijena slika je sada pogodna za izdvajanje ivica pomoću gradijenta, na primer pomoću Zobelovog operatora. Rezultat koji se dobija nakon primene ovog gradijentnog operatora, i nakon toga poređenja sa pragom prikazan je na slici 1.3.3 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici.

Slika 1.3.3. Izdvojene ivice, lena_edges.tif

Važno je napomenuti da se menjanjem standardne devijacije Gausove funkcije, odnosno dimenzija maske filtra, pravi kompromis između dobre lokalizacije ivica (tj. njihove manje debljine) i malog procenta detekcije lažnih ivica. Ukoliko se niskofrekventno filtriranje uopšte ne uradi, dobiće se bolja lokalizacija ivica se na uštrb većeg procenta detekcije lažnih ivica. Radi ilustracije ovog efekta na slici 1.3.4 prikazan je rezultat detekcije ivica kada se prethodno ne izvrši NF filtriranje ulazne slike.

Slika 1.3.4. Izdvojene ivice bez prethodnog NF filtriranja, lena_edges_nofilt.tif

1.4. sign

Na slici 1.4.1 prikazana je ulazna slika iz koje se idvajaju ivice, **sign.tif**.

Slika 1.4.1. Ulazna slika, sign.tif

Kao što se može videti, i ovde je ulazna slika veoma zašumljena aditivnim Gausovim šumom, slično kao i ulazna slika iz odeljka 1.1. Dakle, i ovde je prvi korak poboljšanje kvaliteta slika pomoću niskofrekventnog bilateralnog filtra. Rezultat nakon NF filtriranja pomoću funkcije **bilateral_filter** sa pogodno odabranim parametrima **sigma_s** i **sigma_r** prikazan je na slici 1.4.2.

Slika 1.4.2. Slika nakon NF filtriranja bilateralnim filtrom, sign_bilateral.tif

Slika će nakon ovog filtriranja imati značajno manji procenat detekcija lažnih ivica, a čitljivost natpisa na znaku je očuvana. Ovako dobijena slika je sada pogodnija za izdvajanje ivica nekom od gradijntnih tehnika. Kako je najvažniji sadržaj na slici koji treba izdvojiti sam natpis, odabran je gradijentni operator koji daje najbolje izdvajanje ivica u tekstu. Zbog raznolikosti orijentacija gradijenta u tekstu, odabran je izotropni Fraj-Čenov operator, koji podjednako dobro izdvaja horizontalne, vertikalne i dijagonalne ivice. Rezultat koji se dobija nakon primene ovog gradijentnog operatora, i nakon toga poređenja sa pragom prikazan je na slici 1.4.3 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici. Zbog velike zašumljenosti ulazne slike procenat detekcije lažnih ivica je popriličan, ali je očuvana čitkost značajnih natpisa na znaku.

Slika 1.4.3. Izdvojene ivice, sign_edges.tif

1.5. stop

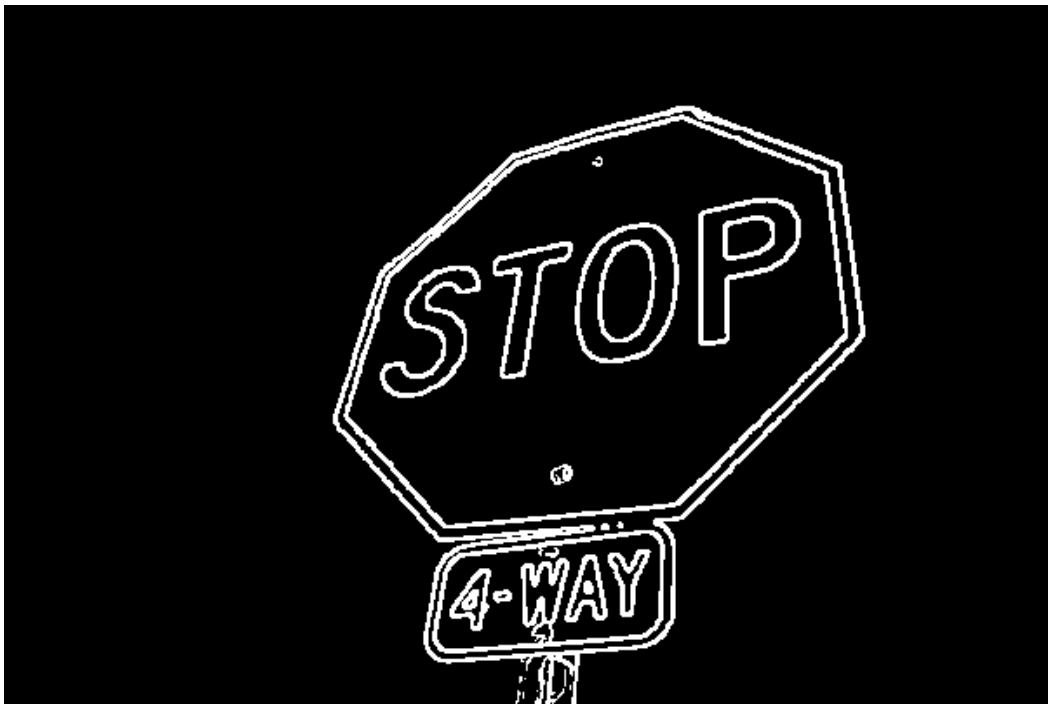
Na slici 1.5.1 prikazana je ulazna slika iz koje se idvajaju ivice, **stop.tif**.

Slika 1.5.1. Ulazna slika, stop.tif

Kao što se može videti, ulazna slika je prilično zašumljena, i to impulsnim šumom. Stoga je prvi korak koji treba napraviti poboljšanje kvaliteta slike u smislu potiskivanja ovog šuma uz što manje uništavanje detalja (oblika znaka i natpisa na njemu). To je pokušano sa nelinearnim filtriranjem pomoću MATLAB funkcija **medfilt2** i **ordfilt2** (sa okolinom u obliku krsta). Najbolje rezultate, u smislu što manjeg zamućenja natpisa na znaku, dala je kombinacija ova dva nelinearna filtriranja. Dakle, dva sukcesivna nelinearna filtriranja pomoću filtara manjeg reda dala su bolje rezultate od pojedinačnih filtriranja pomoću filtara većeg reda. Rezultat je prikazan na slici 1.5.2.

Slika 1.5.2. Slika nakon uklanjanja impulsnog šuma, stop_no_impulse_noise.tif

Ovako dobijena slika je sada pogodna za izdvajanje ivica nekom od gradijntnih tehnika. Pokušano je sa raznim gradijentim operatorima i nema bitnih razlika među njima kada se primene na ovu sliku. Rezultat koji se dobija nakon primene Zobelovog operatora i poređenja sa pragom prikazan je na slici 1.5.3 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici.



Slika 1.5.3. Izdvojene ivice, stop_edges.tif

1.6. van

Na slici 1.6.1 prikazana je ulazna slika iz koje se idvajaju ivice, **van.tif**.

Slika 1.6.1. Ulazna slika, van.tif

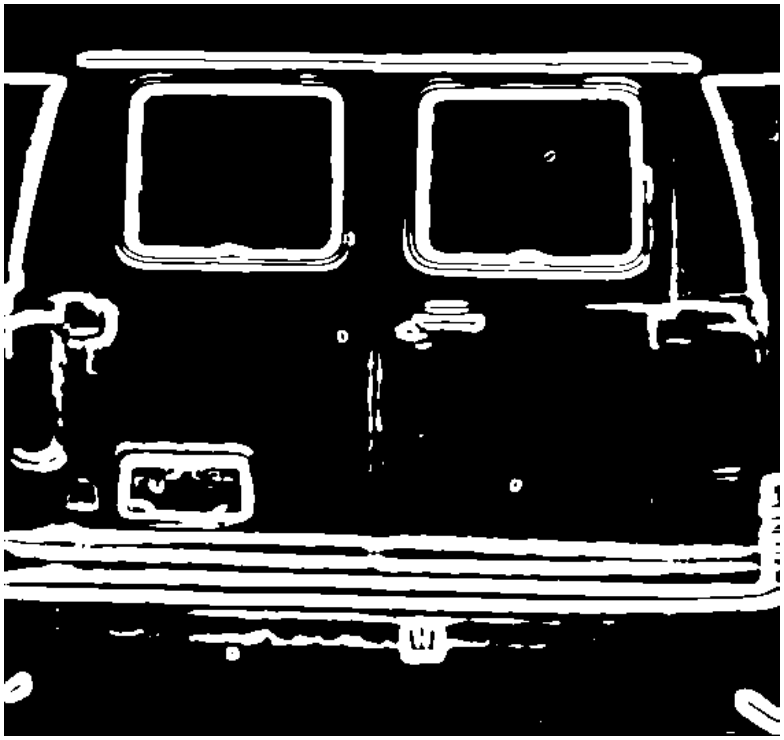
Kao što se može videti, ulazna slika je veoma neuniformno osvetljena, što je degradacija slike koja je multiplikativnog karaktera. Stoga je pogodno primeniti postupak sličan homoforfnom filtriranju. Prvi korak je, dakle, logaritmovanje ulazne slike. Dobijeni rezultat prikazan je na slici 1.6.2.

Slika 1.6.2. Slika nakon logaritmovanja, van_logarithm.tif

Dobijena je slika nešto ujednačenije osvetljenosti i sada je pogodnije izvršiti niskofrekventno filtriranje linearnim Gausovim NF filtrom. Prilikom odabira parametara filtra vodilo se računa o što manjem zamućenju detalja na slici na račun kasnije bolje detekcije ivica. Rezultat nakon NF filtriranja prikazan je na slici 1.6.3.

Slika 1.6.3. Slika nakon NF filtriranja, van_lp.tif

Ovako dobijena slika je sada pogodnija za izdvajanje ivica pomoću nekog gradijentog operatora. U ovom slučaju najbolje rezultate daje operator sa većom maskom filtra, tačnije Abduov piramidalni operator dimanzija 7x7. Zbog većih dimenzija maske, manja je osvetljivost ovog operatora na šum, koji je i dalje u velikoj meri pristutan na slici. Rezultat koji se dobija nakon primene ovog gradijentnog operatora, i nakon toga poređenja sa pragom prikazan je na slici 1.6.4 i predstavlja izlaznu sliku u procesu detekcije ivica u ulaznoj slici.



Slika 1.6.4. Izdvojene ivice, van_edges.tif

Tačka 2: detekcija ivica korišćenjem Kanijevog algoritma

2.1. implementacija

Detekcija ivica pomoću Kanijevog algoritma implementirana je u funkciji **canny_edge_detection**. Na primeru slike **lena.tif** (slika 1.3.1) biće prikazani međurezultati u procesu detekcije ivica. Algoritam se sastoji iz nekoliko koraka koji su u nastavku detaljno opisani.

Najpre, pri ulasku u funkciju, proverava se validnost ulaznih parametara: ulazne slike **I**, standardne devijacije Gausove funkcije **sigma**, kao i vrednosti nižeg praga **GT_low** i višeg praga **GT_high**. Ukoliko su parametri validni, nastavlja se sa realizacijom funkcije.

Prvi korak predstavlja niskofrekventno filtriranje ulazne slike Gausovom funkcijom zadate standardne devijacije. Za veličinu prozora Gausove funkcije se uzima prvi neparni ceo broj veći ili jednak od **6·sigma**. Filtriranje se obavlja u prostornom domenu pomoću MATLAB funkcija **fspecial** i **imfilter**. Rezultat filtriranja sa parametrom **sigma = 1** prikazan je na slici 2.1.1.

Slika 2.1.1. Slika nakon NF filtriranja Gausovom funkcijom, lena_gaussian.tif

Drugi korak predstavlja određivanje horizontalnih i vertikalnih gradijenata filtrirane slike, **Gx** i **Gy**, redom . To se ponovo obavlja pomoću funkcije **imfilter**, uz prethodno specificiranje operatora horizontalnog i vertikalnog gradijenta. Prva dva koraka su ustvari ekvivalentna filtriranju pomoću DroG operatora.

Treći korak predstavlja određivanje magnitude i ugla gradijenata (**Gm** i **Ga**, redom) na osnovu izračunatih horizontalnih i vertikalnih gradijenata. Magnituda gradijenta se računa kao kvadratni koren iz sume kvadrata horizontnog i vertikalnog gradijenta, a ugao gradijenta pomoću MATLAB funkcije **atan2**. Ugao se računa počev od pozitivnog dela "n-ose" (pozitivni deo x-ose u pravouglom Dekartovom koordinatnom sistemu), u smeru kretanja kazaljke na satu. Magnituda gradijenta prikazana je na slici 2.2.2.

Slika 2.1.2. Magnituda gradijenta, lena_gradient_magnitude.tif

Četvrti korak predstavlja kvantizaciju gradijenta na jedan od 4 pravca: **-45°**, **0°**, **45°** i **90°**. To se radi tako što se najpre postojeći opseg ugla gradijenta od **-π** do **π** radijana preskalira na opseg od **-180°** do **180°**. Nakon toga radi se suplementiranje određenih uglova tako da se opseg uglova svede na opseg od **-90°** do **90°**. Ova transformacija ustvari znači da nas kod izdvajanja ivica interesuje samo njihov pravac, a ne i to sa koje njene strane se nalazi svetlija, odnosno tamnija površina (dakle, smer gradijenta). Nakon toga, uglovi se diskretizuju na jedan od 4 pomenuta ugla prema tome kojem od njih su najbliži po pravcu. Ovaj korak je važan preduslov za sledeći korak, koji za cilj ima da poboljša lokalizaciju ivica.

Peti korak predstavlja potiskivanje vrednosti gradijenta koje ne predstavljaju lokalne maksimume (tzv. lokalni ne-maksimumi). Najpre se svakom od pomenuta 4 diskretna pravca pridruži logička matrica (sa vrednostima 0 i 1) koja služi za izdvajanje susednih piksela u diskretizovanom pravcu gradijenta za svaki piksel. To su matrice **H1**, **H2**, **H3** i **H4**, redom. Ideja je da se za svaki piksel u matrici magnitude gradijenta dohvati najpre njegova okolina 3x3 (**neighbourhood**), koja se zatim skalarno pomnoži sa njemu odgovarajućom gore opisanom logičkom matricom. Za tako dobijenu matricu **line**, proverava se da li je trenutni piksel jednak najvećem elementu ove matrice. Ukoliko to jeste slučaj (trenutni piksel jeste lokalni maksimum), u matrici potisnutih lokalnih ne-maksimuma (**Gm_surpressed**) zadržava se vrednost trenutnog piksela. Inače se vrednost ovog piksela potiskuje, odnosno vrednost u izlaznoj matrici mu se postavlja na nulu. Kako bi se ispitivanje za svaki piksel radilo na isti način, pre same obrade radi se proširivanje matrice magnitude gradijenta nulama sa svake strane, čime se dobija proširena matrica magnitude gradijenta, **Gm_ext**. Rezultat nakon potiskivanja lokalnih ne-maksimuma prikazan je na slici 2.1.3.

Slika 2.1.3. Magnituda gradijenta nakon potiskivanja lokalnih ne-maksimuma, lena_gradient_magnitude_surpressed.tif

Šesti korak predstavlja određivanje mapa jakih i slabih ivica na osnovu vrednosti nižeg i višeg praga i magnitude gradijenta sa potisnutim lokalnim ne-maksimumima. Mapa jakih ivica **E_strong** sadrži sve one piksele čija magnituda gradijenta prelazi viši prag, i to su sigurne ivice. Jake ivice prikazane su na slici 2.1.4. Mapa slabih ivica **E_weak** sadrži sve one piksele čija magnituda prelazi samo niži, ali ne i viši prag, i to su potencijalne ivice. Slabe ivice prikazane su na slici 2.1.5.

Slika 2.1.4. Jake ivice, lena_strong_edges.tif

Slika 2.1.5. Slabe ivice, lena_weak_edges.tif

Sedmi i poslednji korak predstavlja uključivanje u mapu ivica onih slabih ivica koje su povezane sa nekom jakom ivicom. Ovo je iterativni postupak gde se u svakoj iteraciji dodaje određeni broj potencijalnih ivica u mapu sigurnih ivica, sve dok je to moguće. Trenutna mapa sigurnih ivica čuva se u matrici **E_ext** i na početku joj se dodeljuje mapa jakih ivica. Trenutna mapa potencijalnih ivica čuva se u matrici **E_candidates_ext** i na početku joj se dodeljuje mapa slabih ivica. Obe ove matrice su proširene nulama iz istog razloga kao što se to radilo i u petom koraku. U svakoj iteraciji najpre se dohvati lista svih nenultih elemenata u mapi potencijalnih ivica pomoću MATLAB funkcije **find**. Zatim se za svaki taj nenulti element dohvata okolina od **8** suseda odgovarajućeg piksela u matrici sigurnih ivica, **neighbourhood**. Ukoliko se u okolini trenutnog piksela nalazi bar jedan nenulti element (sigurna ivica), obeleži se logičkom jedinicom piksel na ogovarajućoj poziciji logičke matrice **addition**, kao piksel koji će biti prebačen iz mape potencijalnih, u mapu sigurnih ivica. Na kraju svake iteracije se logičkim operatorima nad pomenutim matricama **E_ext**, **E_candidates_ext** i **addition** ažuriraju mape potencijalnih i sigurnih ivica. Postupak je završen onda kada nijedna potencijalna ivica više nije susedna ni sa jednom sigurnom ivicom, dakle u iteraciji kada matrica **addition** postane nula matrica. Po završetku iterativnog procesa samo se vrši odsecanje dodatih nula u matrici sigurnih ivica i vraća se rezultat u obliku logičke matrice detektovanih ivica, **E**. Krajnji rezultat prikazan je na slici 2.1.6.

Slika 2.1.6. Izdvojene ivice, lena_canny_edges.tif

Poredeći detekciju ivica iz odeljka 1.3 (slika 1.3.3.) i detekciju ivica pomoću Kanijevog algoritma (slika 2.1.6.) možemo uočiti nekoliko bitnih razlika. Širine ivice kod Kanijevog algoritima su širine **1** piksel (jedinični odziv na ivicu), što je dosta pogodnije za delove slike sa puno detalja. Takođe, pomoću Kanijevog algoritma, odnosno korišćenja dva umesto jednog praga za gradijent, dobijamo bolju kontrolu nad detekcijom lažnih ivica.

2.2. testiranje

U ovom odeljku testirana je funkcija **canny_edge_detection** i na ulaznim slikama **camerman.tif**, **house.tif** i **van.tif**. U nastavku su prikazani međurezultati kao i izlazne slike u procesu detekcije ivica na primeru ove 3 slike. Rezultati za sliku **camerman.tif** prikazani su na slikama od 2.2.1 do 2.2.6, za sliku **house.tif** na slikama od 2.2.7 do 2.2.12, a za sliku **van.tif** na slikama od 2.2.13 do 2.2.18, redom.

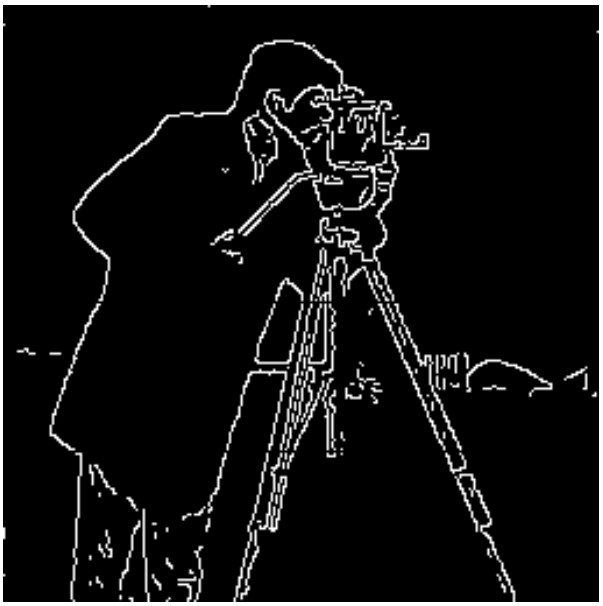
Slika 2.2.1. Slika nakon NF filtriranja Gausovom funkcijom, camerman_gaussian.tif

Slika 2.2.2. Magnituda gradijenta, camerman_gradient_magnitude.tif

Slika 2.2.3. Magnituda gradijenta nakon potiskivanja lokalnih ne-maksimuma, camerman_gradient_magnitude_surpressed.tif

Slika 2.2.4. Jake ivice, camerman_strong_edges.tif

Slika 2.2.5. Slabe ivice, camerman_weak_edges.tif

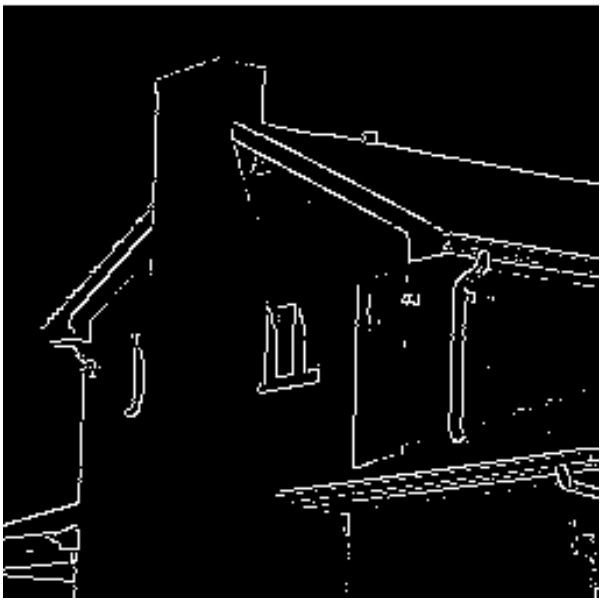


Slika 2.2.6. Izdvojene ivice, camerman_canny_edges.tif

Slika 2.2.7. Slika nakon NF filtriranja Gausovom funkcijom, house_gaussian.tif

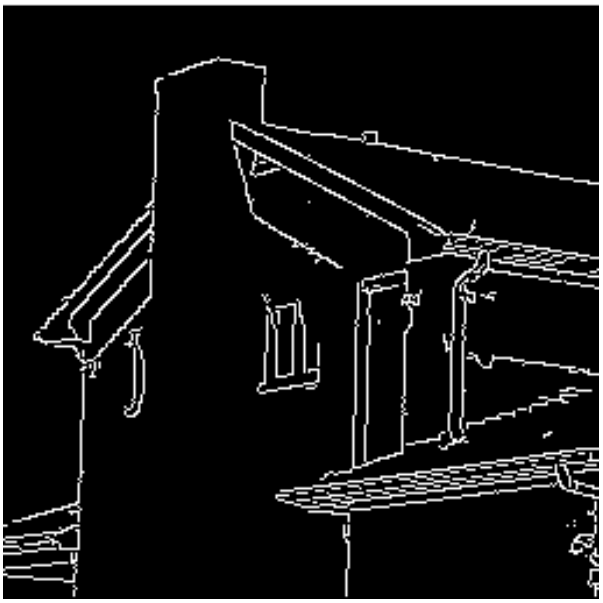
Slika 2.2.8. Magnituda gradijenta, house_gradient_magnitude.tif

Slika 2.2.9. Magnituda gradijenta nakon potiskivanja lokalnih ne-maksimuma, house_gradient_magnitude_surpressed.tif



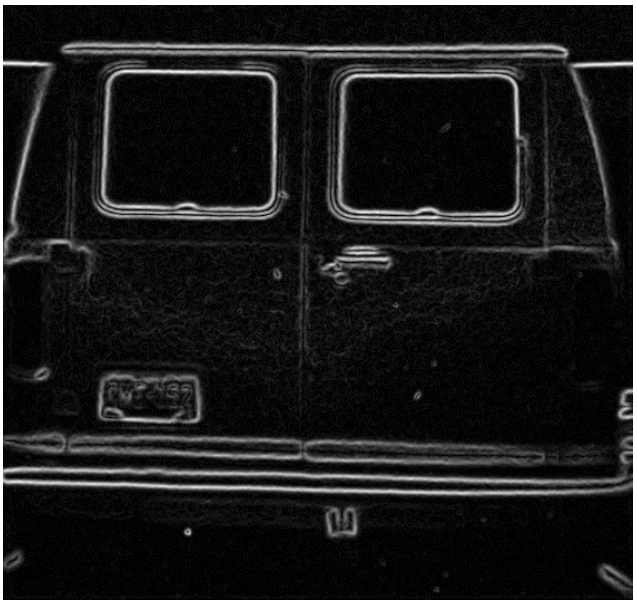
Slika 2.2.10. Jake ivice, house_strong_edges.tif

Slika 2.2.11. Slabe ivice, house_weak_edges.tif



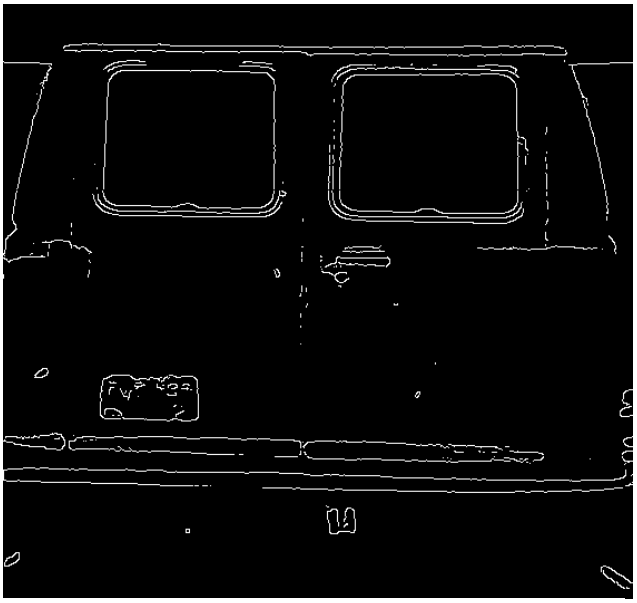
Slika 2.2.12. Izdvojene ivice, house_canny_edges.tif

Slika 2.2.13. Slika nakon NF filtriranja Gausovom funkcijom, van_gaussian.tif

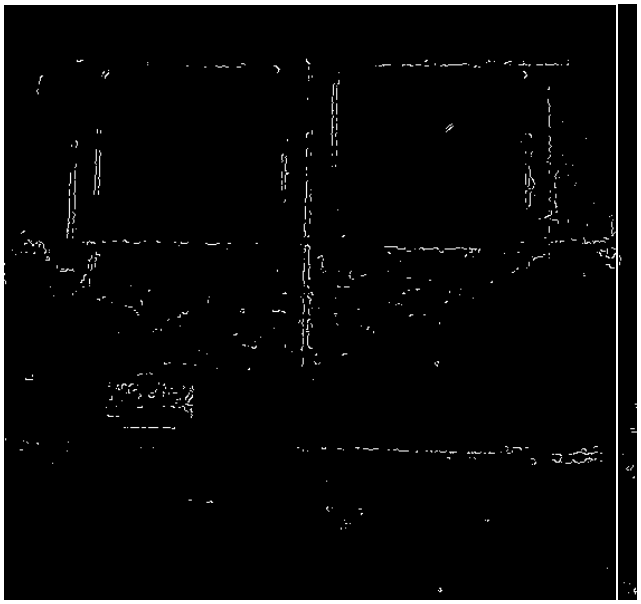


Slika 2.2.14. Magnituda gradijenta, van_gradient_magnitude.tif

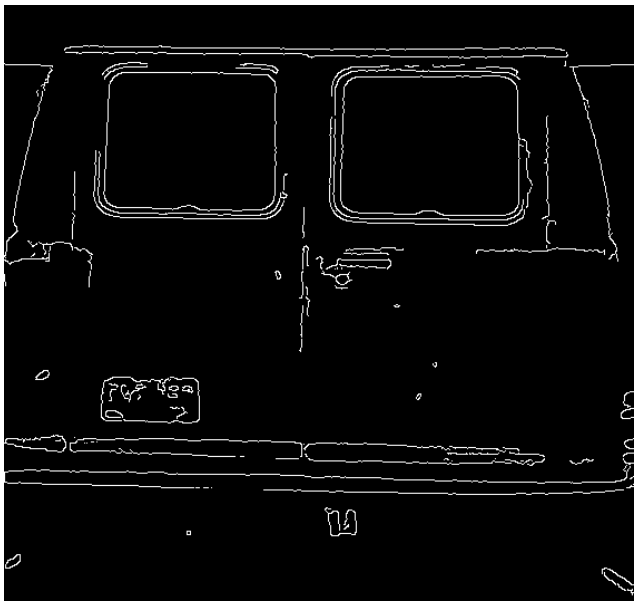
Slika 2.2.15. Magnituda gradijenta nakon potiskivanja lokalnih ne-maksimuma, van_gradient_magnitude_surpressed.tif



Slika 2.2.16. Jake ivice, van_strong_edges.tif



Slika 2.2.17. Slabe ivice, van_weak_edges.tif



Slika 2.2.18. Izdvojene ivice, van_canny_edges.tif

Gore prikazane izdvojene ivice dobijene su podešavanjem ulaznih parametara funkcije **canny_edge_detection** tako da se dobiju što bolji rezultati. Za svaku od slika eksperimentalno su određeni parametri **sigma**, **GT_low** i **GT_high** koji zajedno daju kompromisno rešenje.

Povećavanjem parametra **sigma** u većoj meri uklanjamo šum, tako da se smanjuje procenat detekcije lažnih ivica. Međutim, istovremeno se oštrina pravih ivica smanjuje, te se pogoršava lokalizacija ivica.

Dalje, najpre je za svaku sliku određen odgovarajući viši prag **GT_high** držeći donji prag na vrednosti **1** (dakle, posmatrajući samo sigurne ivice). Viši prag se određuje tako da se izdvoji što veći broj pravih ivica na slici, a da nema mnogo izolovanih ivica, koje uglavnom predstavljaju lažne ivice koje se javljaju usled šuma.

Nakon adekvatno odabranog višeg praga, vrši se podešavanje donjeg praga **GT_low**. Niži prag se određuje tako da se povežu delovi pravih ivica koje su eventualno propuštene sa detekcijom na osnovu višeg praga. Pritom se takođe vodi računa da se ne detektuju izolovane, lažne ivice.

Na slikama 2.2.19, 2.2.20 i 2.2.21 uporedo su prikazani rezultati detekcija ivica za 3 date ulazne slike pomoću metoda iz Tačke 1 i pomoću Kanijevog algoritma.

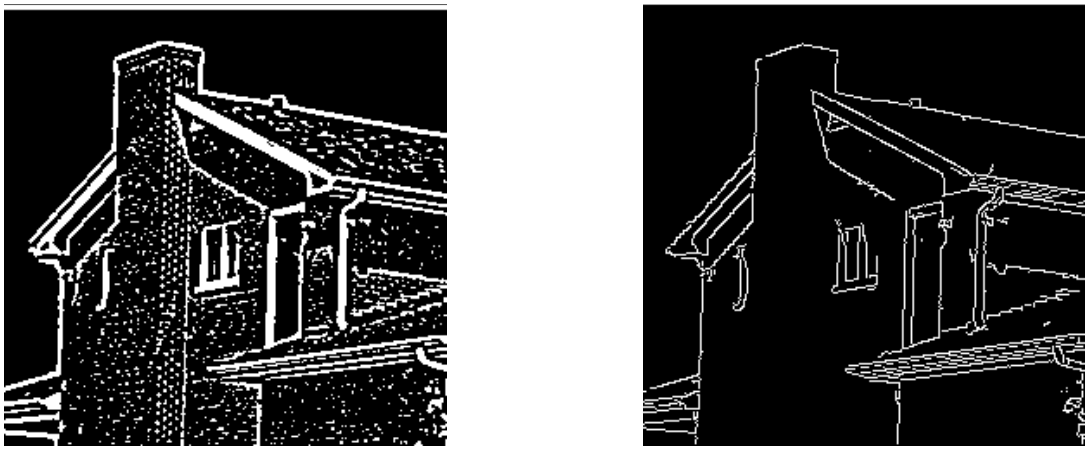
Što se tiče detekcije ivica na slici **camerman.tif**, oba načina detekcije daju vrlo zadovoljavajuće rezultate. Pomoću Kanijevog algoritma bolja je lokalizacija ivica, a manji je i procenat detekcije lažnih, izolavanih ivica. Međutim, broj detektovanih lažnih ivica pomoću metode iz Tačke 1 je zanemarljiv imajući u vidu količinu šuma u polaznoj slici.

Što se tiče detekcije ivica na slici **house.tif**, vidi se prednost detekcije pomoću "agresivnijeg" LoG operatora kada je ulazna slika veoma čista. U tom slučaju, na fasadi i krovu kuće vidljive su čak i ivice koje potiču od ciglica i crepova. To se nije uspelo pomoću detekcije ivica Kanijevim algoritmom jer on sprovodi manje agresivan DroG operator. Sa duge strane, prednost Kanijevog algortima je, naravno, u jediničnom odzivu na ivicu.

Što se tiče detekcije ivica na slici **van.tif**, neuniforma osvetljenost polazne slike je veliki problem ako se korsiti standardno filtriranje linearnim filtrom bez ikakve prethodne obrade, kao što je to slučaj kod detekcije ivica pomoću Kanijevog algoritma. Takođe, detekcija ivica ograničena je i korišćenjem DroG, a ne na primer Zobelovog operatora u sklopu Kanijevog algoritma.

(a) *(b)*

Slika 2.2.19. Izdvojene ivice iz slike camerman.tif: rezultat iz Tačke 1 (a) i pomoću Kanijevog algoritma (b)



(a) *(b)*

Slika 2.2.20. Izdvojene ivice iz slike house.tif: rezultat iz Tačke 1 (a) i pomoću Kanijevog algoritma (b)

(a) *(b)*

Slika 2.2.21. Izdvojene ivice iz slike van.tif: rezultat iz Tačke 1 (a) i pomoću Kanijevog algoritma (b)