Elektrotehnički fakultet Osijek

Poboljšanje slike fotografiranog teksta

Matija Lekić

Osijek, siječanj 2016.

UVOD

Područje obrade kvalitete fotografije široko je područje i zahtjeva složenu i iscrpnu analizu. Ovisno o namjeni fotografije, mijenjaju se i metode obrade iste. Obrada fotografija ljudskog lica zahtijevat će, primjerice, potpuno drugačiji pristup od obrade fotografija pejzaža. Svakako, jedna od zanimljivijih vrsta obrada fotografija jest obrada fotografiranog teksta. Budući da je pisani oblik komunikacije u današnjici gotovo razvijeniji od govornog oblika te se razne varijacije pisanog teksta, od rukopisa, preko teksta na ekranu do teksta ispisanog na pisaču, pojavljuju u svakodnevnom okruženju prosječnog čovjeka i kod sve većeg broja ljudi takav oblik komunikacije postaje preferiraniji od govornog, računalna obrada teksta zaslužuje određenu količinu pažnje i truda.

Kada govorimo o fotografiranom tekstu, posebno je važno primijetiti da je sve veća potreba za što bržim dokumentiranjem željenog teksta. Ljudi današnjeg društva gotovo da nemaju vremena sjesti i prepisati komad teksta, niti skenirati dokument na uređaju predviđenom za to. Najbrži način je fotografirati papir sa željenom informacijom i na taj ju način pohraniti na mobilni uređaj ili u memoriju fotoaparata. Međutim, tako fotografirani tekst često sadrži višak informacije, kao što je okolina lista papira, što može odvlačiti pažnju, pogotovo ako je fotografija u boji. Nadalje, kut iz koje je fotografija uzeta može biti takav da je tekst teže pročitati, čak i uz povećanje slike. Zbog toga je bitno odvojiti dio slike koji sadržava tekst od ostatka slike i prilagoditi ga tako da korisnik može imati uspravan pogled na tekst i da višak informacija iz okoline teksta bude uklonjen. Upravo je to cilj ovog projektnog zadatka.

PREGLED PODRUČJA I PROBLEMATIKE

Kao što je rečeno u uvodu, sve je veća potreba za brzom i jednostavnom pohranom teksta ispisanog na pisaču. Budući da je donedavno jedini način očuvanja teksta bilo korištenje skenera, razvila se potreba za bržim i efikasnijim načinom pohrane teksta na mjestu događaja. Većina populacije koristi pametne telefone sa kamerom dostojne rezolucije. Zbog toga su se razvile razne aplikacije za takve uređaje, kojima je cilj fotografiju teksta urediti na način da taj tekst bude pristupačniji za čitanje i uređivanje od jednostavno fotografiranog teksta, koji uz željenu informaciju sadrži i dosta nepotrebnih ili neželjenih informacija.

Tri najpopularnije aplikacije toga područja za iPhone su „Turbo Scan“, „Scanner Pro“ i „Genius Scan“. Te su aplikacije vrlo slične u načinu na koji obrađuju fotografiju teksta. Posjeduju detekciju rubova bez obzira na kut fotografiranja, poravnanje perspektive te uklanjanje smetnji, kao što su sjene i nepoželjna pozadina. „Turbo Scan“ posjeduje kontrole nad svjetlinom dokumenta, kutom rotacije i izborom boja. Također, ima mogućnost prepoznavanja teksta pod uvjetima slabog osvjetljenja, no zahtijeva više fotografija istog teksta za takvu primjenu. „Scanner Pro“ je aplikacija profesionalnije namjene, jer sadrži opciju potpisivanja fotografiranog dokumenta. Također omogućuje ručno podešavanje detektiranih rubova, ukoliko automatsko podešavanje nije detektiralo željeno područje. Sve tri aplikacije omogućuju spremanje uređenih slika u PDF ili JPEG formatu.

Jedna od sličnih aplikacija Android sustava je „Text Fairy“, koja također prima sliku teksta, ali u potpunosti ostavlja korisniku na odabir područje uređivanja. Nakon što korisnik uredi granice područja uređivanja, aplikacija odredi kontrast slova i pozadine te izravna tekst za uspravan pogled. Dodatna mogućnost je uređivanje teksta, što podrazumijeva kopiranje, brisanje, pisanje i slične akcije. Potrebno je voditi računa o količini osvjetljenja pri fotografiranju te oštrini fotografije. Aplikacija ne prepoznaje rukopis. Dostupan je izvorni kod aplikacije u programskom jeziku C++.

Na ovom području postoji i nekoliko online aplikacija. „To-text“ je online aplikacija koja za učitanu sliku teksta i odabrani jezik prepoznaje fotografirani tekst, izdvaja ga i vraća *.txt* datoteku. Međutim, uvelike ovisi o kvaliteti predane slike, to jest o kvaliteti fotoaparata i fotografa, jer u slučaju slika koje su fotografirane pod određenim kutem, ne prepoznaje svako slovo i kao rezultat vraća neprepoznatljive nizove slova.

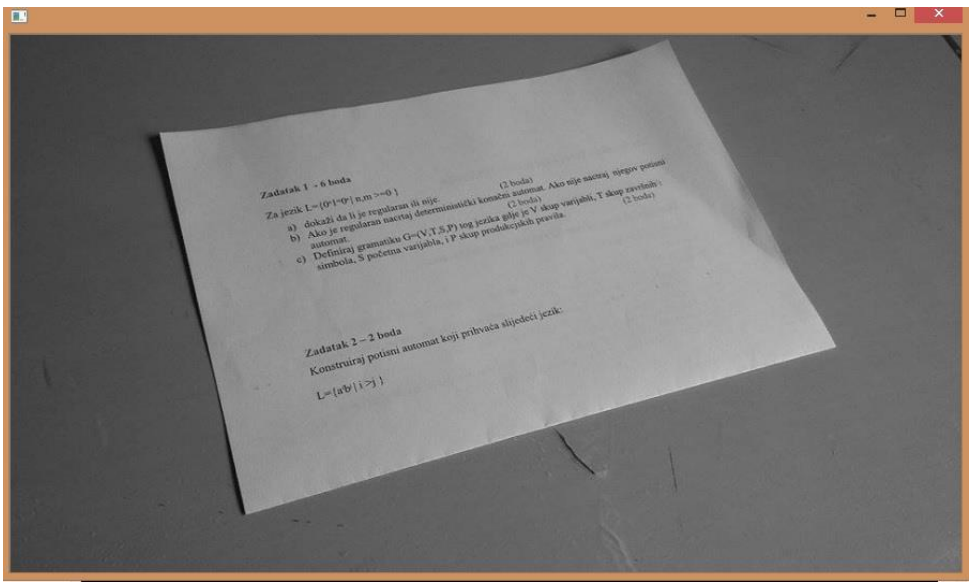
Postoji još nekolicina rješenja na ovom području, ali se u suštini ne razlikuju puno od gore spomenutih. Vidljivo je da svako rješenje kao prvi korak sadržava detekciju rubova, bilo da omogućuje korisniku da odabere područje ili ih prepoznaje sam. Dalje s označenim područjem

radi rotaciju perspektive i ispravlja linije nastale zbog fotografiranja pod određenim kutem. Budući da se očekuje da fotografije stvara korisnik mobilnim uređajem, svaka će slika sadržavati iskrivljene linije i bit će potrebno namjestiti pogled na tekst, tako da ga korisnik može čitati uspravno. Postojeće aplikacije posjeduju mogućnost namještanja kontrasta slova i pozadine ili je kontrast namješten automatski. Opcionalno je prepoznavanje jezika i uređivanje teksta po uzoru na klasične programe za uređivanje teksta. Posljednji korak je spremanje slika ili teksta u odgovarajući format. Po uzoru na navedene mogućnosti i korake u obradi slike fotografiranog teksta, ostvareno je i priloženo rješenje.

OPIS PROBLEMA I RJEŠENJA

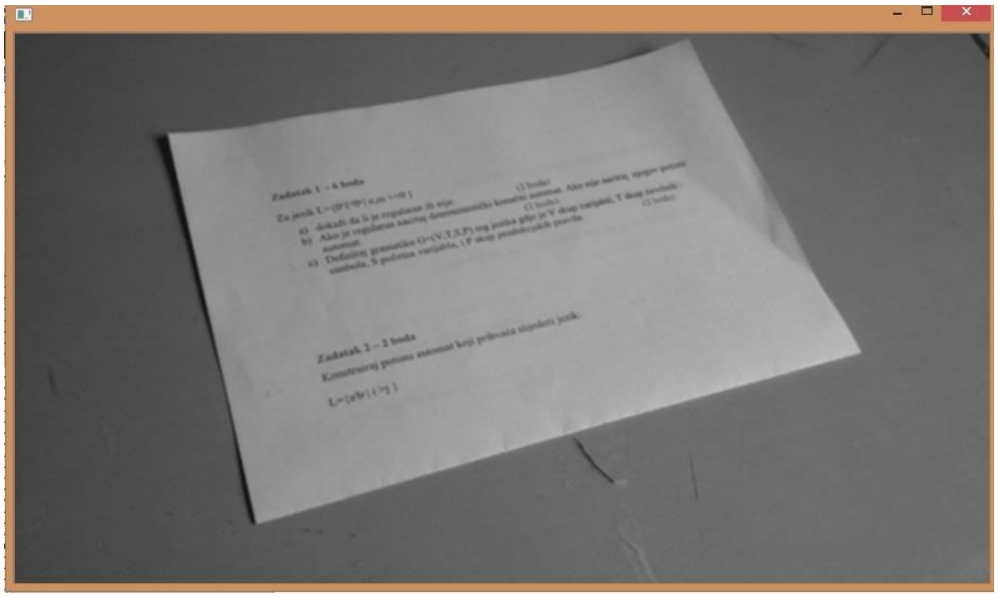
Problem je rješavan u programskom jeziku Python uz upotrebu naredbi iz dodatnih paketa Numpy i OpenCV. Sve korištene funkcije iz OpenCV-a nabrojane su, a poveznice na njihovu iscrpnu dokumentaciju dane su u popisu literature. Cilj programa jest prepoznati fotografirani list papira u odnosu na ostatak slike, to jest, pozadinu. Prepoznati list papira se tada rotira tako da se dobije pogled na tekst pod pravim kutem i primjenjuje se threshold kako bi se uklonio višak boje na slici i istaknula slova u odnosu na pozadinu. Kako bi zadani cilj bio izvediv, potrebno je misliti o načinu fotografiranja. Budući da je kao sastavni dio programa implementirano prepoznavanje rubova, bitno je fotografirati cijeli list papira na kojem se nalazi željeni tekst. Također, važno je voditi računa da se list papira nalazi na podlozi koja se razlikuje u boji od njega samog, kako bi rub bio prepoznatljiv. Bitno je obratiti pažnju na to je li papir savinut na vrhovima ili po sredini, jer u tom slučaju postoji mogućnost zasjenjenja određenih dijelova papira i mogućnost da program prepozna kao rub nešto što nije rub ili da propusti prepoznati postojeći rub.

Prvi dio posla jest na učitanoj slici prepoznati rubove. Prije nego se na slici izvrši prepoznavanje rubova, potrebno joj je smanjiti dimenzije. U ovom slučaju korišteno je nekoliko slika različitih dimenzija, ovisno o uređaju kojim je obavljeno fotografiranje. Međutim, uzeta je vrijednost visine slike od 500 piksela, kako bi se ubrzao proces obrade slike i detektiranje rubova učinilo točnijim. Također, vrijednost visine slike od 500 piksela je vrijednost koja prilikom prikaza slike na zaslonu računala ne prelazi granice radne površine, stoga ju je moguće sagledati u cijelosti. Slika je zatim prebačena iz područja boje BGR u nijanse sive boje (grayscale).



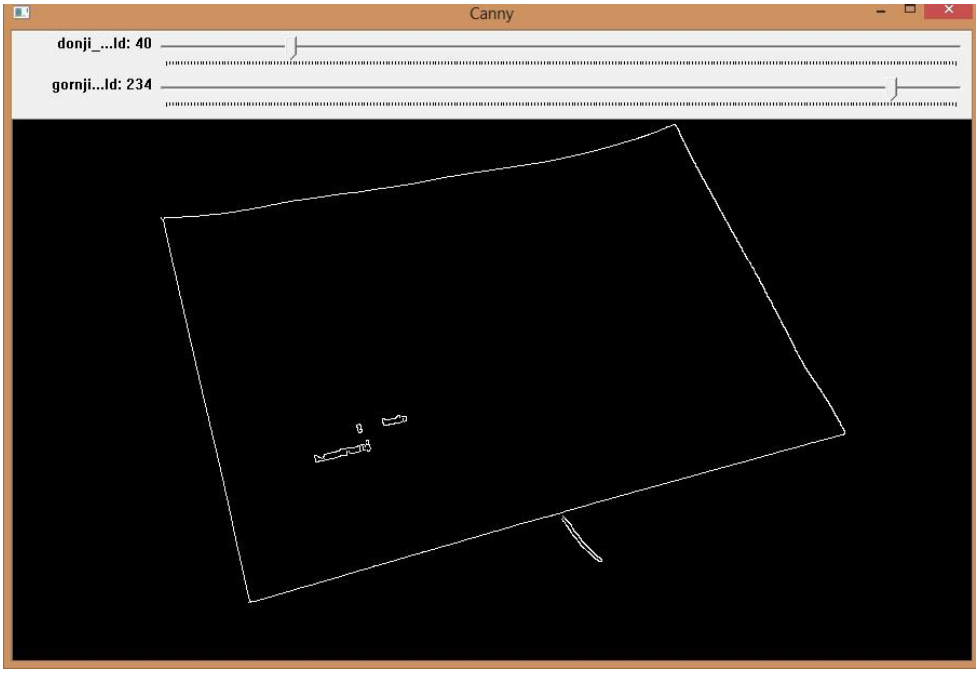
Slika 1. Originalna fotografija u grayscale-u

Kako bi se izgladila slika i uklonili utjecaji šuma na slici, korištena je Gaussova funkcija, odnosno, gausovo zamućenje funkcijom iz OpenCV-a. Pri tome je korištena maska dimenzija 5x5 (prema „R. Gonzales, R. Woods: Digital Image Processing; str. 580).



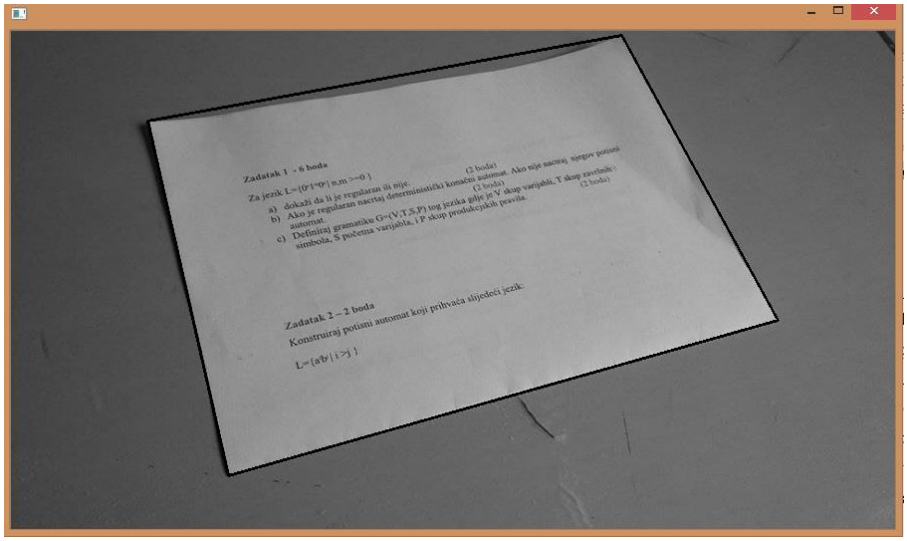
Slika 2. Primijenjeno Gaussovo zamućenje s matricom 5x5

Na takvu zamućenu sliku primijenjen je Canny-ev detektor rubova, koji prima dva parametra, a to su gornji i donji prag, kojima se određuju vrijednosti iznad kojih je element slike prepoznat kao rub. Omogućeno je korisniku birati vrijednosti pragova preko klizača i promatrati kako se mijenja prikaz rubova na slici, s time da je cilj da ostaje sadržan jedino rub koji predstavlja obris papira ili da se barem većina ostalih rubova ukloni.



Slika 3. Podešavanje gornjeg i donjeg praga Canny-evog detektora

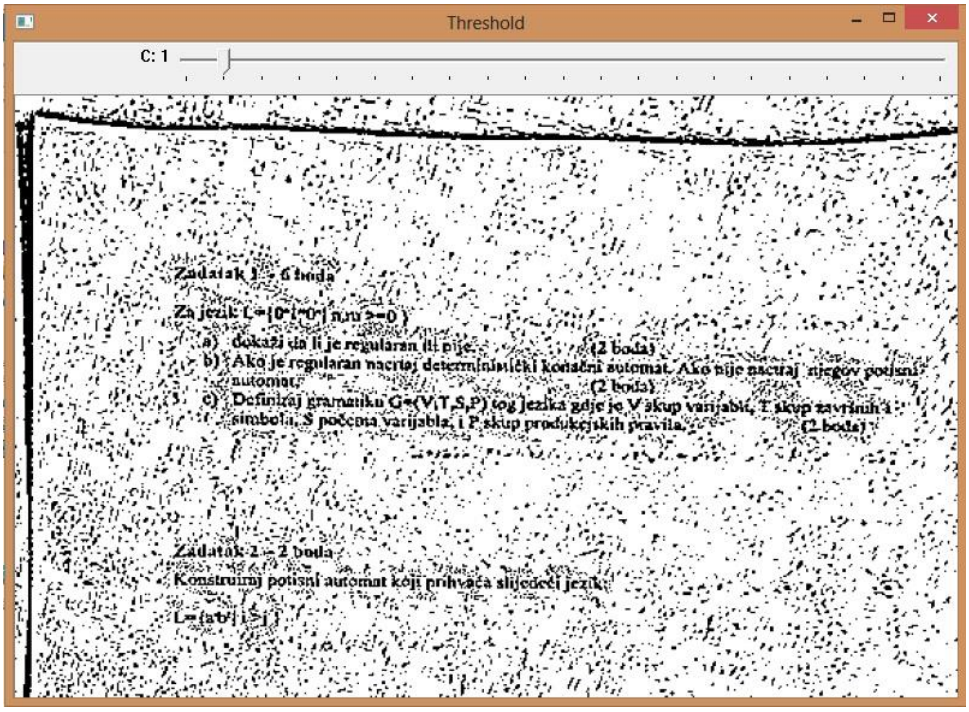
Nakon što su na slici detektirani rubovi i vrijednosti pragova postavljene tako da se istaknu rubovi koji predstavljaju obris papira, potrebno je prepoznati zatvorenu konturu koju sačinjavaju četiri ruba, koji se međusobno dotiču u četiri točke. U ovom koraku do značaja dolazi pažljivo fotografiranje dokumenta, budući da je potrebno fotografirati cijeli papir, tako da program može prepoznati zatvorenu konturu. Također, bitno je da pored papira ne bude drugi objekt kojeg će Canny-ev detektor prepoznati kao zatvorenu konturu većih dimenzija od papira, jer način na koji će program prepoznati konturu papira, a odbaciti bilo koju drugu zatvorenu konturu jest uzimanje najveće konture na slici. Sukladno tome, korištena je funkcija za dohvaćanje svih postojećih kontura na slici, koja kao povratnu vrijednost vraća skup točaka. Zatim je iz skupine kontura odabrana ona najveće površine. Na njoj je izvršena aproksimacija četverokutnog oblika, budući da papir fotografiran nesavršenom ljudskom rukom može odstupati od oblika savršenog četverokuta. Aproksimirani četverokut koji omeđuje papir prikazan je na originalnoj slici u sivim nijansama.



Slika 4. Aproksimirani četverokut koji omeđuje fotografirani papir

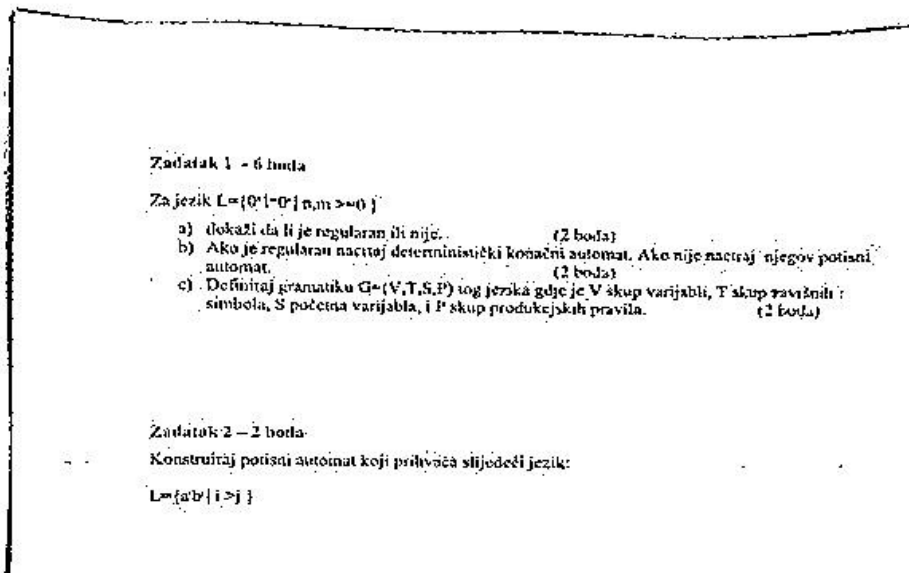
Funkcija koja aproksimira četverokut kao povratnu vrijednost vraća listu četiri točke, koje su vrhovi tog četverokuta. One su nadalje korištene u funkciji za rotaciju pogleda na objekt. Za ispravno daljnje korištenje u spomenutoj funkciji, točke su morale biti posložene u proizvoljnom, ali dosljednom redoslijedu te izračunata širina i visina označenog objekta, jednadžbom udaljenosti između dvije točke. Budući da u pogledu na slici 4 postoji očita razlika u širini papira u gornjem i donjem dijelu te razlika u visini papira u desnom i lijevom dijelu, kao širina, to jest visina, uzeta je veća vrijednost od dvije. Rezultat toga je pojava da mali dio okoline papira ulazi u rezultantnu sliku. Razlog tome je što će dimenzije rezultantne slike biti širina i visina spomenutog četverokuta. Stoga, tamo gdje su širina i visina objekta manje od izračunatih, u područje rezultantne slike će ući okolina objekta, tako da popuni piksele do zadane dimenzije.

Kada je slika rotirana tako da se može pogledati u uspravnom pogledu, preostaje primjena thresholda na nju kako bi se slova istaknula u odnosu na papir te kako bi se eventualne sjene i mrlje na papiru uklonile. Upotrijebljen je adaptivni thresholding i to s Gaussovom jezgrom. Veličina koju korisnik može mijenjati promjenom položaja klizača jest vrijednost konstante koja se oduzima od srednje vrijednosti okoline svakog pojedinog piksela. Mijenjajući tu vrijednost, moguće je fino podesiti sliku i riješiti se nekih smetnji. Optimalna vrijednost te konstante je 10, no potrebno je pratiti što se događa s tekstom, jer se pri uklanjanju šuma mogu ukloniti i bitni pikseli koji sačinjavaju slova.

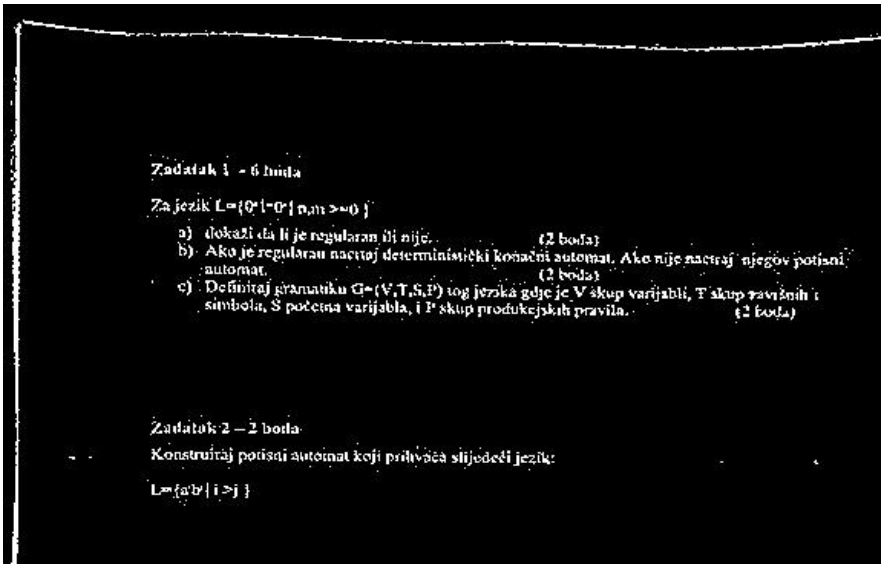


Slika 5. Podešavanje vrijednosti konstante za uklanjanje smetnji

Ovisno o vlastitim preferencama, može se izvršiti klasični binarni threshold ili inverzni binarni threshold, koji će za rezultat dati crnu boju tamo gdje je bijela i obrnuto. Slike se spremaju u JPEG formatu. Rezultante slike za normalni i inverzni threshold su priložene.



Slika 6. Rezultanta slika za normalni threshold



Slika 7. Rezultanta slika za inverzni threshold

Popis funkcija iz OpenCV-a:

cv2.resize

cv2.GaussianBlur

cv2.createTrackbar

cv2.getTrackbarPos

cv2.Canny

cv2.adaptiveThreshold

cv2.getPerspectiveTransform

cv2.warpPerspective

cv2.findContours

cv2.arcLength

cv2.approxPolyDP

cv2.drawContours

ZAKLJUČAK

Uzme li se u obzir postavljeni cilj, program u suštini radi ono što bi trebao raditi; mijenja perspektivu fotografiranog teksta i prikazuje ga kao skenirani dokument u opciji *black and white*. Međutim, rezultat neće biti zadovoljavajuć ako papir nije fotografiran u cijelosti te ako postoji sjena ili bljesak svjetlosti na slici koji bi mogao pokvariti detektiranje rubova ili u konačnici onemogućiti uklanjanje smetnji putem adaptivnog thresholda. Zbog toga neke slike nisu uspjele u tom procesu dati zadovoljavajući rezultat, pod kojim se podrazumijeva čitljiv tekst. To je, dakako, daljnji poticaj da se izvrši poopćenje programa i da se uspije u uklanjanju nastalih problema kako bi program za svaku sliku mogao dati zadovoljavajući ishod. Ovaj je primjer rađen na računalu, no uz daljnji napor, može se preinačiti u mobilnu verziju pa bi rezultanta slika mogla biti dobivena neposredno nakon fotografiranja unutar nekoliko sekundi. Uz ostale slične aplikacije koje postoje na tržištu, zanimljivo je razmišljati o usavršavanju i doradi ovoga projekta kako bi uz punu funkcionalnost mogao konkurirati postojećim rješenjima.

LITERATURA:

- R. C. Gonzales, R. E. Woods: Digital Image Processing. Prentice Hall, Second Edition, 2002.
- Materijali s predavanja kolegija „Obrada slike i računalni vid“, mr. sc. Irena Galić, prof.: Feature Extraction I, Elektrotehnički fakultet Osijek, 2016.
- http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_setup/py_intro/py_intro.html
- http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_geometric_transformations/py_geometric_transformations.html
- http://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html
- http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html
- http://docs.opencv.org/master/da/d22/tutorial_py_canny.html#gsc.tab=0
- https://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_gui/py_trackbar/py_trackbar.html#trackbar
- http://docs.opencv.org/2.4/modules/highgui/doc/user_interface.html
- http://www.tutorialspoint.com/python/python_pass_statement.htm
- http://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html
- http://docs.opencv.org/master/dd/d49/tutorial_py_contour_features.html#gsc.tab=0
- http://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html
- http://www.to-text.net
- https://play.google.com
- https://itunes.apple.com

POPIS PROGRAMA:

„img_text.py“

Program zahtijeva instaliran Python, verzija 2.7. i dodatne datotečne module OpenCV, verzija 2.4.ili novija te Numpy, verzija 1.10. ili novija. Aplikacija se pokreće preko „command prompt“ prozora ili „terminal“, ovisno o operacijskom sustavu. Korisnik mora *cd* naredbama kao trenutni radni direktorij postaviti onaj u kojem se nalazi *.py* datoteka, zatim naredbom *python img_text.py* pokreće program, koji se nakon obavljenog posla sam zatvori.