Prepoznavanje odigranih poteza tijekom šahovske partije

Martinović, Ivan

Master's thesis / Diplomski rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:890522

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

PREPOZNAVANJE ODIGRANIH POTEZA TIJEKOM ŠAHOVSKE PARTIJE

Diplomski rad

Ivan Martinović

Osijek, 2016.

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 11.09.2016.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Ivan Martinović
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D-601R, 23.10.2013.
OIB studenta:	87465582998
Mentor:	Prof.dr.sc. Željko Hocenski
Sumentor:	Doc.dr.sc. Ivan Aleksi
Predsjednik Povjerenstva:	Doc.dr.sc. Ivan Aleksi
Član Povjerenstva:	Doc.dr.sc. Tomislav Matić
Naslov diplomskog rada:	Prepoznavanje odigranih poteza tijekom šahovske partije
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	U ovom radu se na temelju video snimke odigrane 3D šahovske partije prepoznaju odigrani potezi. Primjenom metoda obrade slike potrebno je prepoznati odigrane poteze. Implementaciju metode potrebno je napraviti u MATLAB programskom okruženju. sumentor dr Aleksi
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 3 Jasnoća pismenog izražavanja: 3 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	11.09.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:



IZJAVA O ORIGINALNOSTI RADA

Osijek, 20.09.2016.

Ime i prezime studenta:	Ivan Martinović
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D-601R, 23.10.2013.
Ephorus podudaranje [%]:	0%

Ovom izjavom izjavljujem da je rad pod nazivom**: Prepoznavanje odigranih poteza tijekom šahovske partije**

izrađen pod vodstvom mentora Prof.dr.sc. Željko Hocenski

i sumentora Doc.dr.sc. Ivan Aleksi

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD **1**

2. MATLAB PROGRAMSKO OKRUŽENJE **2**

2.1. MATLAB skriptni jezik 2

2.2. Izgled i kompatibilnost MATLAB-a 3

3. REALIZACIJA PROGRAMA **5**

3.1. Stvaranje i priprema video snimke šahovske partije 5

3.2. Priprema slike za obradu 7

3.3. Obrada slike 8

3.3.1. Podešavanje kontrasta slike 8

3.3.2. Stvaranje sive i binarne slike 13

3.3.3. Stvaranje integralne slike 15

3.4. Obrada podataka 16

3.5. Detekcija odigranih poteza 18

3.6. Ispis odigranih poteza 21

4. EKSPERIMENTALNI REZULTATI **24**

5. ZAKLJUČAK **29**

LITERATURA **30**

SAŽETAK **31**

ABSTRACT **32**

ŽIVOTOPIS **33**

PRILOZI **34**

1. UVOD

Cilj diplomskog rada je izraditi program u MATLAB programskom okruženju koji će na temelju video snimke odigrane 3D šahovske partije prepoznati i na odgovarajući način ispisati odigrane poteze. Prepoznavanje odigranih poteza vrši se primjenom metoda obrade slike unutar MATLAB programskog okruženja. Kako bi se navedeni program realizirao, potrebno je 3D problem šahovske partije svesti na 2D problem nakon čega se vrši konkretna obrada slike. Rad se sastoji od dva osnovna poglavlja, poglavlja u kojem su opisane karakteristike, mogućnosti te prednosti i nedostaci korištenja MATLAB programskog okruženja i poglavlja u kojem je opisana realizacija programa navedenog problema. Kroz rad razmotrit će se razne metode obrade slike, mogućnosti primjene pojedinih metoda, njihove prednosti i nedostaci kao i razlozi korištenja navedenih metoda.

2. MATLAB PROGRAMSKO OKRUŽENJE

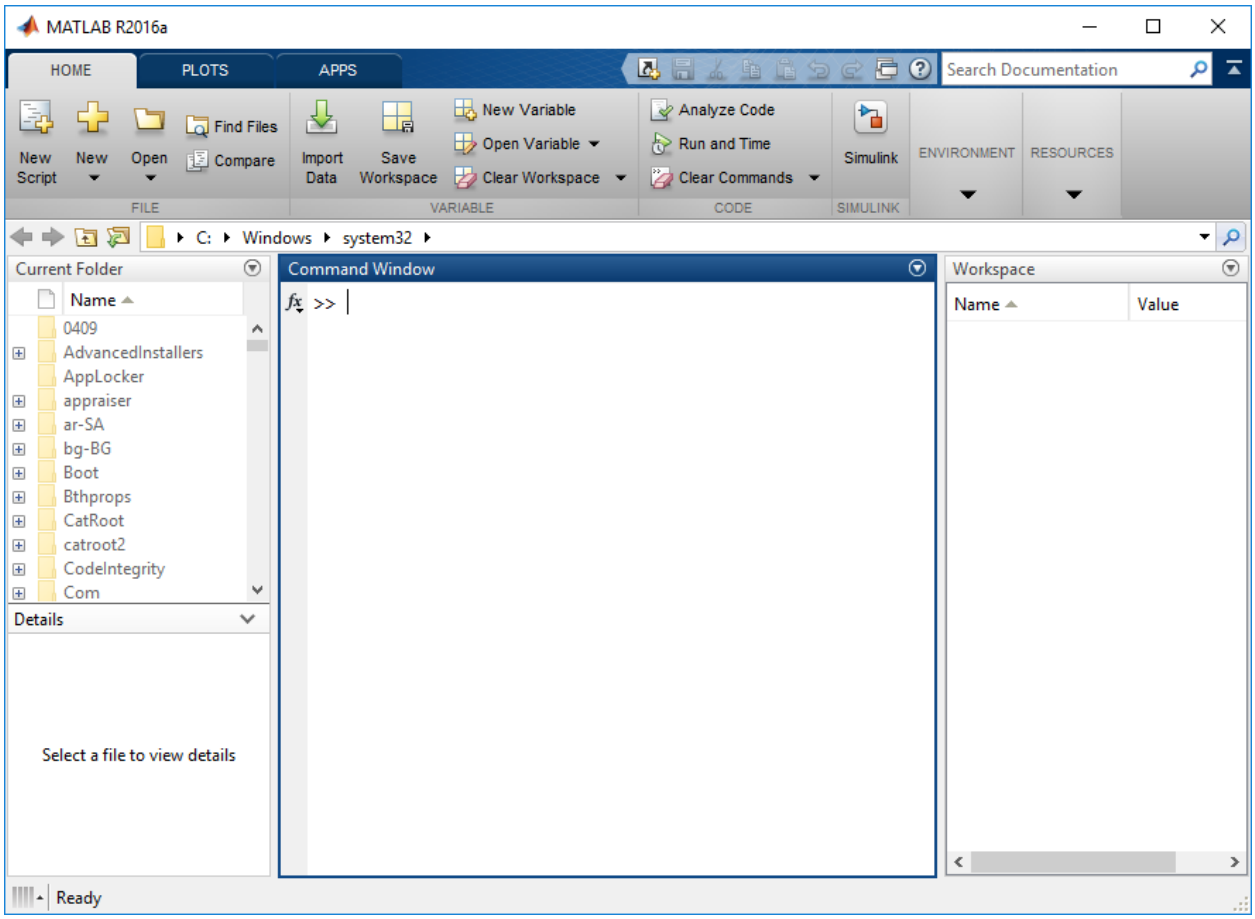
MATLAB (engl. ***mat****rix* ***lab****oratory*) je programski jezik četvrte generacije razvijen od strane američke korporacije *MathWorks* koja se specijalizira za izradu programske podrške za matematičke proračune. MATLAB omogućava manipulaciju matricama, grafički prikaz funkcija i podataka, implementaciju algoritama, izradu vlastitog sučelja, mogućnosti obrade slike i brojne druge mogućnosti. Kratki vodič kroz osnove programiranja i obrade slike u MATLAB programskom okruženju možete pronaći u [1].

2.1. MATLAB skriptni jezik

MATLAB skriptni jezik često se uspoređuje s C++, Java i Python programskim jezicima. MATLAB je skriptni jezik koji je izrađen isključivo s ciljem olakšavanja i ubrzavanja matematičkih i drugih proračuna dok su C++ i Python programski jezici opće namjene iako u današnje vrijeme postoje razne biblioteke koje omogućavaju slične ili iste mogućnosti. Prednosti MATLAB skriptnog jezika u odnosu na druge programske jezike uključuju efikasnost proračuna s matricama i poljima, jednostavan način prikaza i vizualizacije podataka, proširivost (dostupnost raznih alata – *toolbox*), jednostavan način upravljanja datotekama, jednostavan način obrade nizova, nema potrebe za deklariranjem varijabli – sve što je potrebno je njihova inicijalizacija.

Iako ima brojne prednosti, MATLAB skriptni jezik ima i nekoliko iznimno bitnih nedostataka u odnosu na ostale objektno-orijentirane jezike. Vjerojatno najbitniji nedostatak odnosi se na brzinu izvođenja MATLAB skripti. Može se pokazati da je za rješavanje istog problema MATLAB i do nekoliko puta sporiji u odnosu na druge programske jezike (prema [2]). Kako bi se ovaj nedostatak izbjegao može se provesti paralelizacija koda ili je pak moguće pribjeći korištenju GPU biblioteka (CUDA, OpenCV, arrayFire i dr.) te izvršavati kod na grafičkoj kartici. U tom slučaju potrebno je dodati nešto više linija koda budući da se sve varijable i podaci, nad kojima se vrši obrada, moraju prebaciti na grafičku karticu te zatim nakon završetka obrade i obavljenih proračuna natrag u radni prostor MATLAB-a.

2.2. Izgled i kompatibilnost MATLAB-a



Slika 2.1. *MATLAB programsko okruženje.*

Na slici 2.1. prikazan je osnovni izgled MATLAB programskog okruženja. Najbitniji dijelovi grafičkog sučelja su komandni prozor (engl. *command window*) i radni prostor (engl. *workspace*). Komandni prozor jedan je od glavnih alata za unos podataka, izvršavanje MATLAB funkcija i M-datoteka kao i za prikaz dobivenih matematičkih rezultata. Grafički rezultati (grafovi, histogrami i sl.) prikazuju se u odgovarajućem skočnom prozoru. Radni prostor omogućava prikaz i manipulaciju svim kreiranim varijablama koje su direktno postavljene ili izračunate tijekom MATLAB sesije. Od ostalih bitnijih dijelova MATLAB grafičkog sučelja može se istaknuti i povijest naredbi (engl. *command history*) koja omogućava brzi pristup prethodno pozivanim naredbama i linijama koda (do najviše 25 000) te trenutni direktorij (engl. *current directory*) koji omogućava brzi pristup datotekama u trenutnom radnom direktoriju MATLAB-a.

Također, treba napomenuti da je jedna od velikih prednosti MATLAB-a kompatibilnost verzija. Osim standardne kompatibilnosti, odnosno mogućnosti pokretanja programa kreiranih u nekoj od starijih verzija u novijoj verziji, MATLAB omogućava i pokretanje programa kreiranog u novijoj verziji u bilo kojoj starijoj verziji bez potrebe za dodatnim modifikacijama koda kao što je to primjerice potrebno pri korištenju *MS Visual Studio* programskog okruženja gdje je potrebno u zaglavlju glavne programske datoteke promijeniti verziju u kojoj se program pokreće. Naravno, postoje slučajevi gdje se određene funkcije neće moći pokrenuti ako su uvedene isključivo u novijoj verziji što se može lako popraviti dodavanjem M-datoteka u direktorij MATLAB-a ukoliko su datoteke, kojima su definirane navedene funkcije, dostupne za besplatno preuzimanje putem službenih stranica. U suprotnom potrebno je izvršiti nadogradnju sustava.

3. REALIZACIJA PROGRAMA

Kako bi se program za detekciju i ispisivanje odigranih šahovskih poteza realizirao, potrebno je cjelokupni problem podijeliti u što više cjelina te zatim navedene cjeline detaljno razraditi. Osnovni algoritam za rješavanje ovog problema može se podijeliti u sljedeće cjeline:

1. Učitavanje video snimke odigrane šahovske partije
2. Priprema slike za obradu
3. Obrada slike
4. Ispis odigranih poteza.

Kako bi se program mogao izvršavati u stvarnom vremenu (u idealnom slučaju trenutni okvir video snimke trebao bi se obraditi prije nego stigne slijedeći okvir) uvedena su razna pojednostavljenja od načina snimanja odigrane šahovske partije pa sve do načina na koji se odigrani potezi ispisuju.

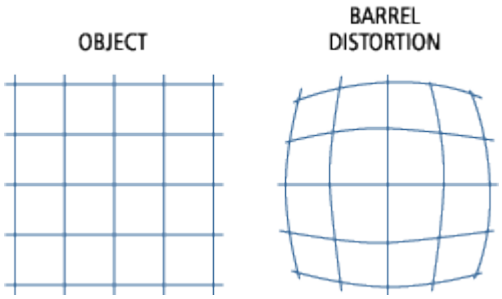
3.1. Stvaranje i priprema video snimke šahovske partije

Stvaranje video snimke šahovske partije moguće je izvršiti korištenjem bilo kojeg uređaja za snimanje video zapisa. Za potrebe testiranja realiziranog diplomskog rada snimljen je video zapis korištenjem mobitela u rezoluciji 640×480 piksela uz 30 slika (okvira) u sekundi. Primjer slike ovog video zapisa prikazan je na slici 3.1.



Slika 3.1. *Isječak iz video snimke šahovske partije.*

Na temelju slike 3.1. može se vidjeti da je odabrani način snimanja šahovske partije iz ptičje perspektive pri čemu je kamera pozicionirana iznad središta šahovske ploče uz približno jednaku udaljenost od svih rubova ploče. Ovakav način snimanja odabran je radi pojednostavljenja složenosti problema. Svaka figura sadržana je unutar jednog polja i ne dolazi do preklapanja polja i figura. Na slici također možemo vidjeti da je došlo do neželjene pojave iskrivljenja slike, tzv. efekta bačve (engl. *barrel effect*). Do ove pojave dolazi kod leća sa širokim kutom[1]. Naime, vidno polje leće znatno je šire od veličine senzora slike pa se stoga slika mora „stisnuti". Slika 3.2. detaljnije prikazuje ovu pojavu.



Slika 3.2. *Iskrivljenje slike zbog svojstava leće.*

Izvor: *http://umich.edu/~lowbrows/guide/opticaljargon.html.*

Kako bi se umanjio utjecaj efekta bačve, video zapis je potrebno obraditi prije učitavanja što je moguće realizirati primjenom programa poput *DeFishr* i *Adobe After Effects*, koji nažalost nisu besplatni te je potrebno kupiti odgovarajuću licencu[2]. Alternativni način je primjena dostupnih besplatnih biblioteka unutar MATLAB programskog okruženja (vidi [3]).

Učitavanje video zapisa u MATLAB programsko okruženje omogućeno je korištenjem funkcije *VideoReader* kojoj se kao argument predaje ime video zapisa na slijedeći način (uz pretpostavku da se video zapis nalazi u trenutnom radnom direktoriju MATLAB programskog okruženja; u suprotnom je potrebno navesti apsolutnu putanju):

```
video = VideoReader('ime_datoteke');
```

[1] leće čija je žarišna duljina znatno manja nego kod normalnih leća
[2] Licenca navedenih programa kreće se od 100$ do 500$

Izvođenjem navedene linije koda stvara se objekt imena *video* koji sadržava sve informacije o učitanom video zapisu (ukupno trajanje, razlučivost, ukupni broj slika, ime, oznake i dr.) kao što je prikazano na slici 3.3.

```
video =

  VideoReader with properties:

   General Properties:
            Name: 'sah.mp4'
            Path: 'C:\Users\Ivan Martinović\Desktop\Diplomski rad'
        Duration: 39.5730
     CurrentTime: 0
             Tag: ''
        UserData: []

   Video Properties:
           Width: 640
          Height: 480
       FrameRate: 30
    BitsPerPixel: 24
     VideoFormat: 'RGB24'
```

Slika 3.3. *Prikaz informacija o učitanom video zapisu.*

3.2. Priprema slike za obradu

Kako bi se učitala pojedina slika iz prethodno stvorenog objekta *video* koristi se funkcija *read()* kojoj se kao argument predaje ime objekta, koji sadrži podatke o video snimci, i redni broj slike koja se želi učitati. Kao rezultat nastaje matrica veličine $M \times N \times 3$ gdje su M i N broj redaka i stupaca matrice koji direktno ovise o razlučivosti slike. Primjerice, ako je učitana slika veličine 640×480 tada će nastala matrica biti također dimenzija $640 \times 480 \times 3$. Matrica je trodimenzionalna jer se radi o slici u boji (R, G i B dimenzija).

Nakon učitavanja željene slike vrši se promjena dimenzija učitane slike s ciljem ubrzavanja obrade. Bitno je napomenuti da se ovim postupkom ne smanjuje kvaliteta slike. Promjena dimenzija izvodi se pozivom funkcije *imresize()* na slijedeći način:

```
Slika = imresize(ime_matrice, [veličina_slike], metoda_interpolacije);
```

Za određivanje elemenata nove matrice *Slika* koristi se jedna od metoda interpolacije: bilinearna (za određivanje intenziteta na određenom mjestu promatraju se pikseli u okolnom 2×2 susjedstvu), bikvadratna (za određivanje intenziteta na određenom mjestu promatraju se pikseli u okolnom 4×4 susjedstvu) ili metoda najbližih susjeda (za određivanje intenziteta na određenom mjestu promatra se samo vrijednost najbližeg susjeda). Ukoliko se navedena metoda ne specificira, tada se automatski odabire bikvadratna metoda čije je vrijeme izvođenja ujedno i najduže. Želimo li primjerice što preciznije i kvalitetnije rezultate tada je najbolje koristiti bikvadratnu metodu. U tom slučaju kao posljedica dolazi do usporavanja brzine izvođenja programa. Razlog korištenja ove funkcije je utjecanje na broj elemenata matrice nad kojima će biti potrebno provoditi ostatak proračuna. Ako uzmemo konkretan slučaj definiran ovim diplomskim radom, sliku veličine $640 \times 480 \times 3$ (921 600 elemenata) svodimo na sliku veličine $480 \times 480 \times 3$ (691 200 elemenata) čime smo smanjili broj elemenata za gotovo jednu trećinu bez subjektivnog smanjenja kvalitete novonastale slike.

Prepoznavanje ploče ostvareno je na način da se od korisnika prilikom pokretanja programa zahtijeva unos četiri točke koje su ujedno i rubne točke šahovske ploče. Ovim postupkom znatno je unaprijeđeno vrijeme obrade jer ne postoji potreba za korištenjem dodatnih algoritama i metoda obrade slike. Primjerice, ako bi koristili postojeću funkciju *detectCheckerboardPoints()*, unutar MATLAB programskog okruženja, za otkrivanje uzorka šahovske ploče vrijeme izvođenja za svaku sliku bi se povećalo za približno 0.2 do 0.3 sekunde. Ako navedeno vrijeme usporedimo s vremenom u kojem se treba obraditi svaka pojedinačna slika ($\frac{1}{30}$, odnosno $0.0\dot{3}$ sekunde) može se zaključiti kako bi obrada slike u realnom vremenu već u ovom dijelu bila nemoguća. Dodatni problem nastaje jer bi se u tom slučaju navedene točke, dobivene kao rezultat izvođenja funkcije, morale i sortirati prije korištenja.

3.3. Obrada slike

Postupak obrade slike, kao i postupak realizacije programa, sastoji se iz nekoliko koraka:

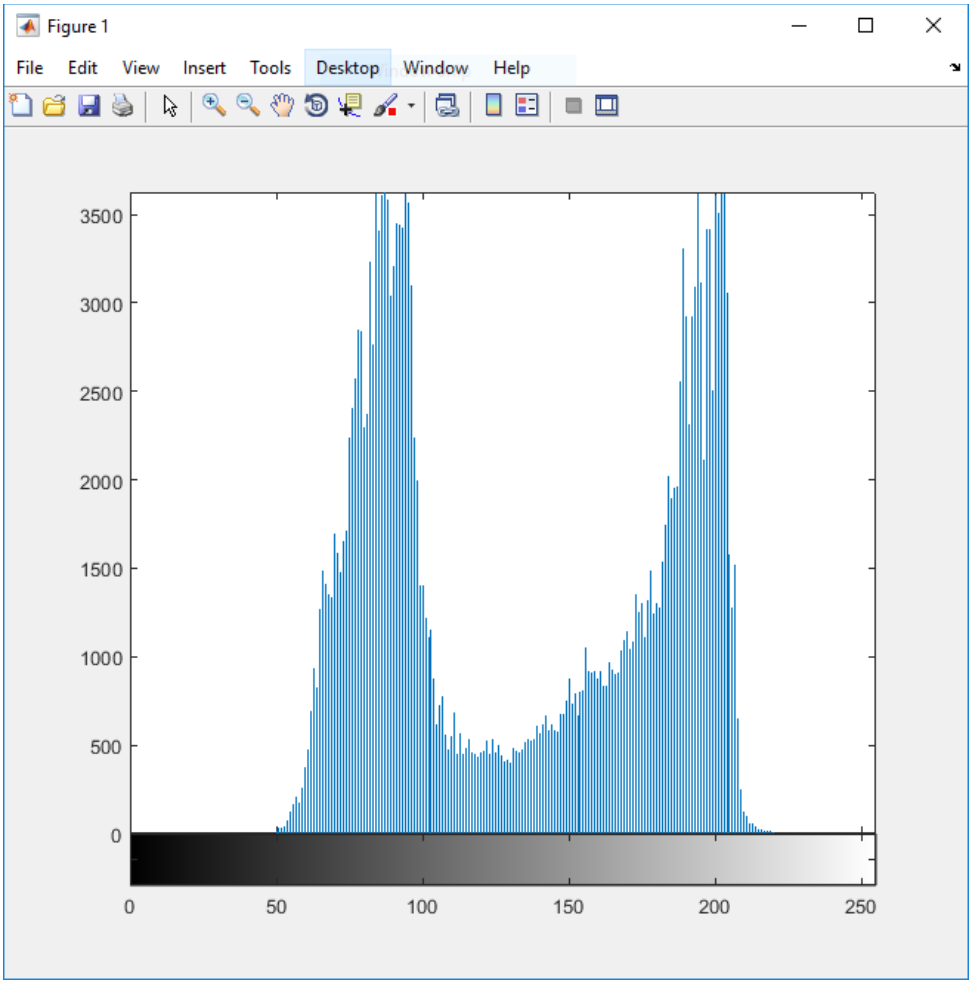
1. Podešavanje kontrasta slike
2. Stvaranje sive i binarne slike
3. Stvaranje integralne slike.

Svi prethodno navedeni koraci biti će detaljno objašnjeni u nastavku ovog poglavlja zajedno s prednostima i nedostacima svih korištenih metoda. Ovisno o kvaliteti slike dodatno se može provesti i izoštravanje slike. Više informacija o načinu izoštravanja slike možete pronaći u [4].

3.3.1. Podešavanje kontrasta slike

Kontrast slike važan je atribut kvalitete slike. Utječe na našu sposobnost spoznaje svjetline i detalja slike, odnosno predstavlja odnos između bijelih i crnih dijelova slike ili jednostavnije rečeno koliko se i objekti međusobno razlikuju. Iako na prvi pogled možda i ne izgleda kao da može imati veliki utjecaj na obradu slike u nastavku poglavlja pokazat će se upravo suprotno.

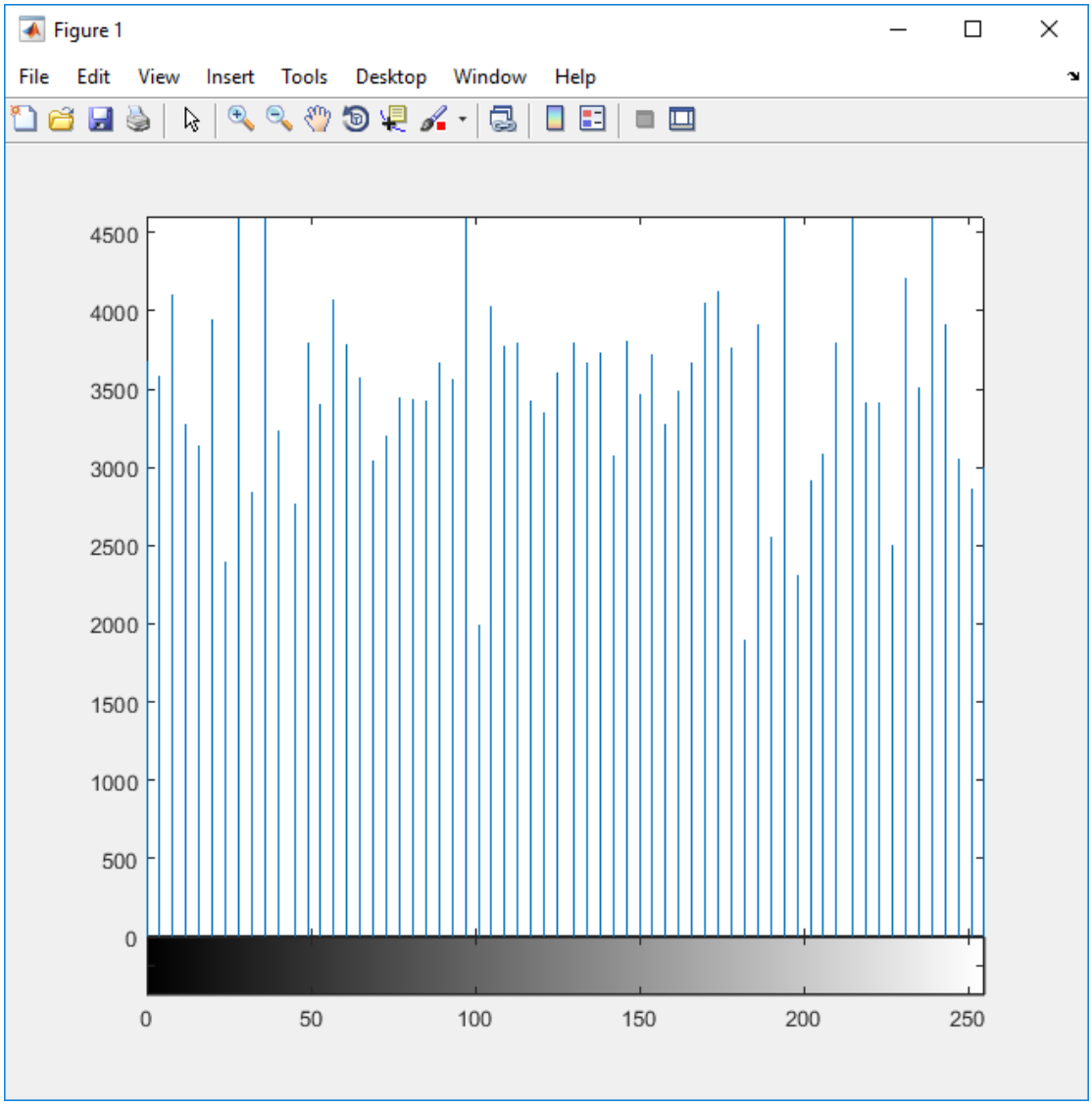
Kako bi se postiglo što veće poboljšanje slike koriste se histogrami. Histogram je stupčasti grafikon koji kao i svaki drugi grafikon ima dvije osi. Stvaranje histograma omogućeno je korištenjem funkcije *imhist()*. Budući da se radi o slici u boji, potrebno je stvoriti tri zasebna histograma za svaki kanal. Izgled histograma učitane slike video zapisa za kanal crvene boje prikazan je na slici 3.4.



Slika 3.4. *Histogram – crveni kanal.*

Iz histograma možemo vidjeti da je na osi x (os apscisa) prikazan intenzitet boje (MATLAB automatski pretvara svaku sliku u sivu sliku prilikom poziva ove funkcije te je zbog toga na osi apscisa prikazan raspon crne i bijele boje) dok je na osi y (os ordinata) prikazan ukupan broj piksela koji imaju isti intenzitet. Ukoliko je na histogramu najveći broj piksela koncentriran isključivo na lijevoj ili isključivo na desnoj strani tada se radi o lošem kontrastu slike, odnosno slika je ili pretamna ili presvijetla te ju je u tom slučaju potrebno korigirati. U idealnom slučaju svi intenziteti slike trebali bi biti jednako zastupljeni. U ovom slučaju može se vidjeti da postoje dijelovi slike koji su pretamni, ali i dijelovi slike koji su presvijetli u odnosu na ostatak slike. Korekcija kontrasta slike u MATLAB programskom okruženju provodi se korištenjem funkcije *histeq()* koja provodi postupak izjednačavanja histograma. Izjednačavanje histograma nastoji histogram izravnati što je više moguće s ciljem postizanja boljeg kontrasta. Postupak se zasniva na transformaciji vrijednosti intenziteta svakog kanala slike zasebno, budući da se radi o slici u boji, na takav način da dobivena

slika odgovara histogramu postavljenom pri pozivu funkcije *histeq()*. To znači da se pri pozivu ove funkcije može specificirati koliki će se broj intenziteta koristiti u histogramu na koje će se svoditi sve ostale vrijednosti intenziteta. Ukoliko se ovaj parametar ne specificira tada se koriste 64 vrijednosti intenziteta. Treba napomenuti da što je veći broj vrijednosti to će i slika ponovno poprimati lošiji kontrast jer će se histogram ponovno svoditi na svoju izvornu verziju. Konkretno u ovom slučaju ukoliko bi se kao parametar postavilo približno 170 vrijednosti intenziteta tada bi korigirani histogram imao identičan izgled kao i histogram prikazan na slici 3.4.



Slika 3.5. *Korigirani histogram – crveni kanal.*

Na slici 3.5. prikazan je korigirani histogram nakon poziva funkcije *histeq()*. Usporedi li se ova slika s prethodnom, možemo doći do zaključka da su u ovom slučaju relativno jednako zastupljene sve vrijednosti intenziteta dok su u prethodnom slučaju značajno prevladavale vrijednosti intenziteta od 70 do 100 te od 170 do 200. Kako bi se pokazalo da je kontrast slike uistinu bitan element pri obradi slike treba pogledati što se događa pri pretvaranju ovako obrađenih slika u binarne (Otsu metoda određivanja praga).



Slika 3.6. *Prikaz binarne slike prije (lijevo) i poslije korekcije kontrasta (desno).*

Na temelju slike 3.6. može se doći do zaključka da je nakon korekcije kontrasta i binarizacije slike došlo do prikaza puno više detalja. Slika u boji izgubila je na svom vizualnom izgledu jer se izjednačavanje histograma provodilo za svaki kanal posebno (metoda se inače provodi na sivoj slici).

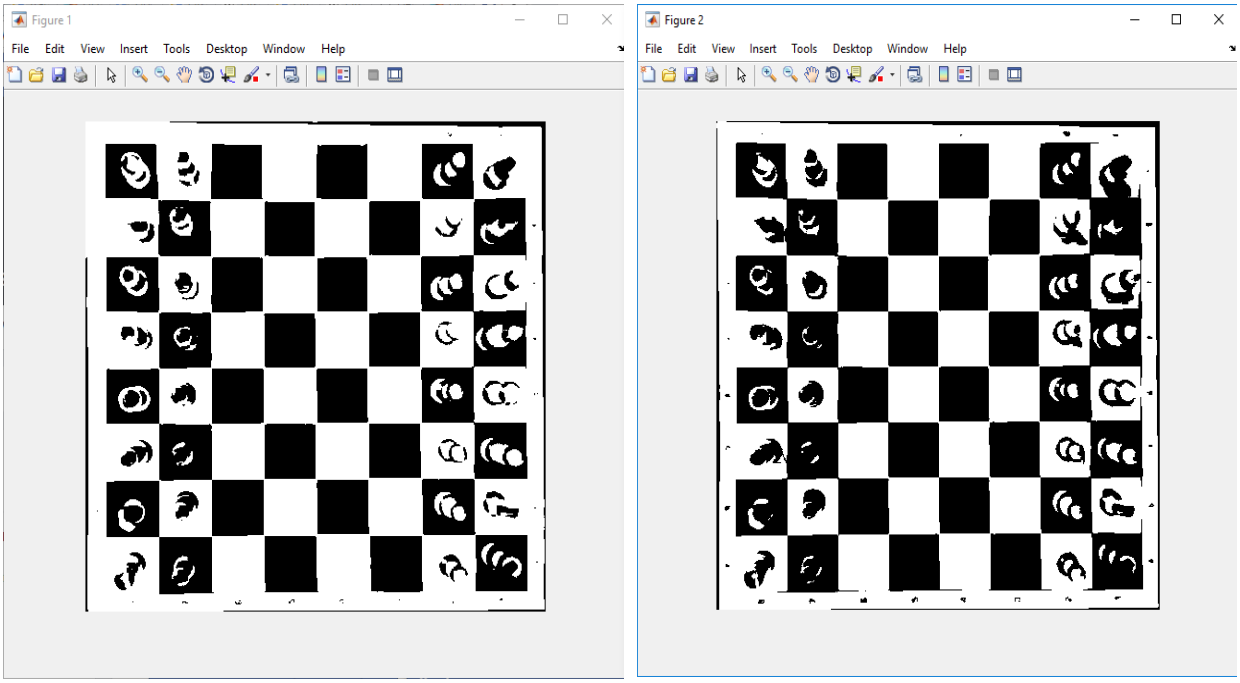
3.3.2. Stvaranje sive i binarne slike

Za potrebe obrade slike najčešće se koristi binarna slika jer je moguće izdvojiti dijelove slike ili piksele od interesa. Na primjer, želimo li na nekoj slici prepoznati crvenu boju, tada možemo specificirati raspon intenziteta boja koji su bitni (npr. 255:0:0 u RGB zapisu ako želimo pravu crvenu boju) te zatim isti koristiti kao prag za stvaranje binarne slike. Tako nastala binarna slika imala bi vrijednost jedan (bijela boja) na mjestima gdje je postojala crvena boja, odnosno nula na mjestima gdje nije bilo crvene boje. Na ovaj način izvršila se detekcija crvene boje. Isti princip primjenjuje se u ovom slučaju za prepoznavanje šahovske ploče i šahovskih figura uz manje promijene. Da bi se slika pretvorila u binarnu prvo treba stvoriti sivu sliku (engl. *grayscale image*). Kako MATLAB funkcija za stvaranje binarne slike ima mogućnost samostalnog pretvaranja slike u sivu, nije potrebno dodavati dodatne linije koda za pretvorbu. Funkcija za stvaranje binarne slike, *im2bw()* koristi se na slijedeći način:

```
BinarnaSlika = im2bw(matrica_slike, vrijednost_praga);
```

Može se vidjeti da navedena funkcija prima dva argumenta. Prvi argument je matrica slike koju želimo pretvoriti u binarnu sliku, odnosno u ovom slučaju to je matrica slike nad kojom je prethodno provedena promjena dimenzija i korekcija kontrasta, dok je drugi argument vrijednost praga koji predstavlja granicu između nule i jedinice. Navedena granica ovisi o samoj slici i ne mora nužno biti aritmetička sredina (0.5). Vrijednost praga najbolje je odrediti Otsu metodom. Otsu metoda je dobila ime po izumitelju prof. Nobuyuki Otsu i koristi se u većini programa koji se bave izvedbama računalnog vida i obrade slike. Metoda polazi od pretpostavke da slika koja će biti podvrgnuta binarizaciji sadrži dvije klase piksela (histogram u dva dijela – primjerice pozadina slike i istaknuti segmenti) te zatim izračunava optimalni prag razdvajanja te dvije klase na način da se unutar klasna varijanca između bijelih i crnih piksela minimizira. Za pronalaženje vrijednosti praga Otsu metodom koristi se funkcija *graythresh()* koja prima samo jedan argument odnosno matricu intenziteta slike. U ovom slučaju dobivena vrijednost praga iznosi 0.498. Glavni razlog

tome leži upravo u činjenici da je nad ovom slikom prethodno izvršena korekcija kontrasta čija je glavna zadaća „izjednačavanje“ svijetlih i tamnih dijelova. Ukoliko se binarizacija slike provede prije provođenja korekcije kontrasta do izražaja dolaze brojne prednosti korištenja Otsu metode, konkretno za ovaj slučaj obrade slike.



Slika 3.7. *Binarizacija slike s default pragom (lijevo) i pragom određenim Otsu metodom (desno).*

Na temelju slike 3.7. možemo vidjeti da je binarizacija slike uz prag određen Otsu metodom puno uspješnija. Prag određen Otsu metodom iznosi 0.5294 dok je na slici lijevo prag iznosio 0.5 budući da nije bio specificiran pri pozivu funkcije *im2bw()*. Na slici desno prikazano je puno više detalja i rubovi su znatno izraženiji. Naravno postoje i slučajevi gdje se pokazalo upravo suprotno. Pojava sjene kod korištenja Otsu metode zapravo je iznimno dobar slučaj jer omogućava da se polja na kojima se nalaze šahovske figure značajnije razlikuju od ostalih polja na kojima se ne nalaze šahovske figure. Iako ima svoje prednosti, Otsu metoda treba se koristiti s oprezom jer značajno ovisi o osvjetljenju i količini svijetlih i tamnih objekata na slici. Ukoliko se ne koristi u kombinaciji s elementima korekcije kontrasta slike vrlo vjerojatno će se pokazati kao neupotrebljiv način odabira praga jer će davati nejasne pa čak i neželjene rezultate. U tom slučaju ručni odabir praga vjerojatno će se pokazati kao najbolja metoda određivanja praga iako će zahtijevati puno više vremena.

Alternativna metoda binarizacije slike može se također zasnivati i na prepoznavanju rubova. Način prepoznavanja šahovskih figura i odigranih šahovskih poteza ovom metodom možete vidjeti u [5].

3.3.3. Stvaranje integralne slike

Nakon realizacije svih prethodnih dijelova, od podešavanja dimenzija slike, podešavanja kontrasta slike pa sve do binarizacije slike preostaje proces stvaranja integralne slike. Integralna slika prvi put se uvodi davne 1984. godine, ali sve do 21. stoljeća nije imala značajniju primjenu u području računalnog vida, točnije do 2001. godine (prema [6]). Integralna slika koristi se za brzo i efikasno određivanje sume intenziteta svih piksela za zadanu sliku. Postupak određivanja sume prilično je jednostavan. Ako imamo primjerice neku točku s koordinatama (x, y) tada će vrijednost koju će integralna slika poprimiti na tom mjestu biti određena prema formuli (3-1):

$$s(x,y) = i(x,y) + s(x-1,y) + s(x,y-1) - s(x-1,y-1) \tag{3-1}$$

gdje je $s(x,y)$ vrijednost integralne slike na mjestu (x,y), $i(x,y)$ intenzitet piksela na mjestu (x,y), $s(x-1,y)$ vrijednost integralne slike na jednom mjestu u lijevo od $s(x,y)$, $s(x,y-1)$ vrijednost integralne slike na mjestu iznad $s(x,y)$ te $s(x-1,y-1)$ vrijednost integralne slike dijagonalno iznad mjesta $s(x,y)$. Da bi sam postupak bio nešto jasniji, promotrimo matricu A s elementima prikazanim na slici 3.8.

```
A =

     1     2     3     4
     5     6     7     8
     9    10    11    12
    13    14    15    16
```

Slika 3.8. *Matrica A.*

Matrica integralne slike sadrži prvi redak i prvi stupac popunjen s nulama. Želimo li tada odrediti elemente matrice integralne slike primjenom formule (3-1) slijedi:

$$s(1,1) = i(1,1) + s(0,1) + s(1,0) - s(0,0) = 1 + 0 + 0 - 0 = 1$$

$$s(1,2) = i(1,2) + s(0,2) + s(1,1) - s(0,1) = 2 + 0 + 1 - 0 = 3$$

$s(2,1) = i(2,1) + s(1,1) + s(2,0) - s(1,0) = 5 + 1 + 0 - 0 = 6$ ...

Nakon rješavanja cjelokupnog postupka slijedi konačno rješenje prikazano na slici 3.9.

```
IntegralnaSlika =

     0     0     0     0     0
     0     1     3     6    10
     0     6    14    24    36
     0    15    33    54    78
     0    28    60    96   136
```

Slika 3.9. *Integralna slika matrice A.*

U MATLAB-u je moguće na više načina odrediti integralnu sliku. Prvi i najsporiji način, koji zahtjeva i najmanje programiranja, je pozivanje funkcije *integralImage()* kojoj se kao argument predaje matrica slike čiju integralnu sliku želimo odrediti. Drugi način, ujedno i vremenski učinkovitiji, je korištenje funkcije *cumsum()* u slijedećem obliku:

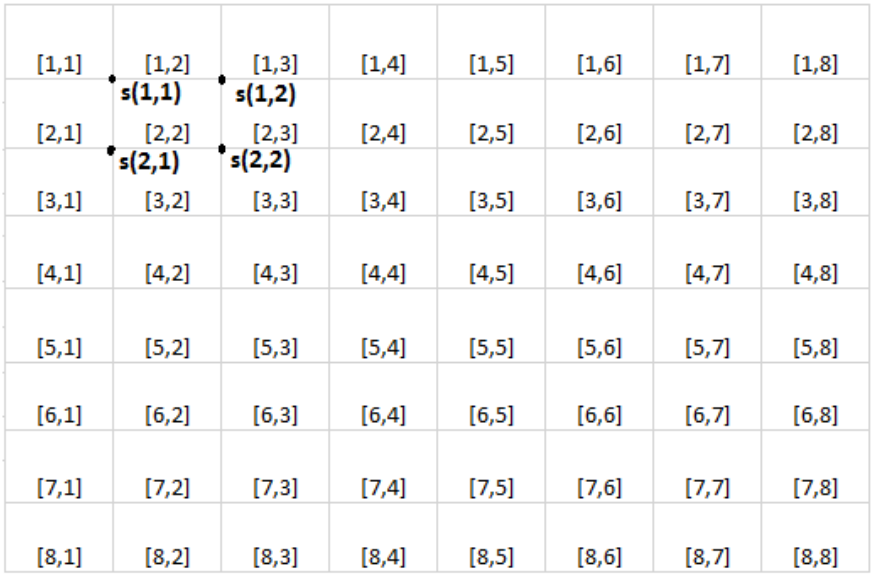
```
integralnaSlika = cumsum(cumsum(double(BinarnaSlika),2))
```

Funkcija *cumsum()* provodi kumulativnu sumu po jednoj specificiranoj dimenziji. Budući da integralna slika predstavlja sumu svih piksela, sumu je potrebno realizirati po obje dimenzije te se stoga funkcija *cumsum()* poziva dva puta, jednom za sumiranje po recima i jednom za sumiranje po stupcima. Osnovna razlika između ovog načina određivanja integralne slike i korištenja funkcije *integralImage(),* osim prethodno navedenih razlika, je što u ovom slučaju ne postoje dodani prvi redak i prvi stupac koji trebaju sadržavati nule. Usporedbe radi, prosječno vrijeme izvođenja funkcije *integralImage()*, odnosno vrijeme određivanja integralne slike za 1000 učitanih slika iznosi 0.00628 sekundi dok je prosječno vrijeme određivanja integralne slike korištenjem funkcije *cumsum()* za isti broj slika 0.00171 sekundi što je razlika od gotovo četiri puta.

3.4. Obrada podataka

Nakon uspješne realizacije integralne slike daljnji postupak prepoznavanja i ispisivanja odigranih poteza zasniva se na razumijevanju podataka prikazanih integralnom slikom. Budući da su prethodno dimenzije učitane slike svedene na kvadratni oblik (480×480 piksela) daljnji

postupak obrade podataka znatno je olakšan. Ono što je prvo potrebno napraviti je podijeliti tako dobivenu integralnu sliku na 64 jednaka dijela (64 šahovska polja na šahovskoj ploči). Ovom podjelom dolazi se do integralne slike za svako pojedinačno šahovsko polje. Kako bi se uklonile moguće smetnje nastale rubnim crtama, efektom bačve i sl., moguće je po potrebi iz računskog postupka izbaciti nekoliko rubnih piksela svakog polja. U ovom slučaju odabrana su četiri rubna piksela svakog polja čime su dimenzije svakog polja smanjene s 60×60 na 56×56 piksela. U prethodnom poglavlju spomenuto je da integralna slika sadrži sumu svih intenziteta piksela do određenog mjesta te stoga nije potrebno ponovno vršiti sumaciju već samo iskoristiti postojeće sume. Tako primjerice, ukoliko iz postojeće integralne slike treba doći do integralne slike za deseto polje s koordinatama polja $(2,2)$, tada od ukupne sume do polja $(2,3)$ trebamo oduzeti sumu piksela prethodnog polja $(1,2)$, sumu piksela polja $(2,1)$ koje se nalazi iznad te dodati sumu piksela polja $(1,1)$ koje se nalazi dijagonalno iznad. Postupak je identičan određivanju integralne slike prema prethodno spomenutoj formuli (3-1). Na slici 3.10. grafički je prikazan navedeni postupak u nešto jednostavnijem obliku.



Slika 3.10. *Koordinate šahovskih polja.*

Prema prethodno napisanom tekstu slijedi:

$$integralnaSlikaPolja(2,2) = s(2,2) - s(2,1) - s(1,2) + s(1,1)$$

Razlog pribrajanja sume intenziteta piksela za prvo polje je zbog toga što sume intenziteta polja s koordinatama $(1,2)$ i $(2,1)$ u sebi već sadrže sumu intenziteta piksela polja s koordinatama $(1,1)$ te se u tom slučaju navedeno polje dvaput oduzima (umjesto jednom). Ukoliko se konkretno uzmu koordinate integralne slike (uz korekciju rubova) tada će formula za polje s koordinatama $(2,2)$ poprimiti oblik:

$$integralnaSlikaPolja(2,2) = integralnaSlika(116,116) - integralnaSlika(116,64) - integralnaSlika(64,116) + integralna\ slika(64,64)$$

Primjenom istog principa dobiju se i integralne slike za preostala šahovska polja.

3.5. Detekcija odigranih poteza

Nakon realizacije matrice koja sadrži sumu intenziteta piksela pojedinih polja prelazi se na realizaciju zadnjeg i najbitnijeg dijela rada, odnosno na detekciju i ispisivanje odigranih poteza tijekom šahovske partije. Prvo je potrebno realizirati matricu koja će pratiti položaj figura tokom partije. Da bi se ta matrica mogla realizirati potrebno je odrediti početni položaj figura. Početni položaj figura može se realizirati na način da se za prvu učitanu sliku izvrši usporedba integralnih slika dvaju polja kako bi se odredilo je li položaj figura vodoravan ili horizontalan, odnosno jesu li figure smještene u recima ili stupcima. Uz matricu igre, koja će pratiti položaje figura, potrebno je realizirati matricu koja će sadržavati odgovarajuću notaciju polja od a1 pa sve do h8. Kako u MATLAB-u nije moguće u matricu direktno spremati znakove i nizove, koristi se naredba *cell()*. Kako bi se iz ćelije pročitala vrijednost na određenom mjestu potrebno je koristiti naredbu *cell2mat()*. Navedeni postupak prikazan je na slici 3.11.

```
>> notacijaPolja =cell({'a8','b8','c8','d8','e8','f8','g8','h8';'a7','b7','c7','d7','e7','f7','g7','h7';...
                        'a6','b6','c6','d6','e6','f6','g6','h6';'a5','b5','c5','d5','e5','f5','g5','h5';...
                        'a4','b4','c4','d4','e4','f4','g4','h4';'a3','b3','c3','d3','e3','f3','g3','h3';...
                        'a2','b2','c2','d2','e2','f2','g2','h2';'a1','b1','c1','d1','e1','f1','g1','h1'});
>> s1 = cell2mat(notacijaPolja(1,1))

s1 =

a8
```

Slika 3.11. *Korištenje naredbi cell() i cell2mat().*

Prepoznavanje poteza odvija se na način da se uspoređuju vrijednosti polja trenutne i prethodno učitane slike uz dozvoljeno odstupanje od maksimalno $\pm 1\%$. Ukoliko je došlo do većeg odstupanja inkrementira se brojač. Da bi potez bio valjan, vrijednost brojača smije poprimiti vrijednosti jedan ili dva. Za sve ostale vrijednosti brojača cjelokupni daljnji postupak se preskače te se prelazi na slijedeću sliku video zapisa. Ukoliko je vrijednost brojača poprimila vrijednost jedan ili dva tada se prelazi u daljnji postupak obrade podataka. Usporedbom dvaju susjednih polja s ploče omogućava se podjela šahovske ploče na bijela i crna polja radi zasebne obrade. Tako na primjer ako za prvu učitanu sliku usporedimo vrijednosti polja s koordinatama $(4,4)$ i $(5,4)$, ili bilo koja dva šahovska polja koja su na početku šahovske partije slobodna, dobijemo da je polje $(4,4)$ bijelo polje ukoliko je vrijednost sume intenziteta piksela veće od sume intenziteta piksela polja $(5,4)$ i obrnuto. Dobivene matrice za bijela i posebno za crna polja dalje normaliziramo. Normalizacija se vrši na način da se cjelokupna matrica podijeli s najvećim elementom te matrice. Odnosno:

```
crnaPolja = crnaPolja/max(max(crnaPolja));
bijelaPolja = bijelaPolja/max(max(bijelaPolja));
```

Naredba *max()* kao rezultat vraća najveći element niza. Budući da se u ovom slučaju radi o matricama, potrebno je funkciju *max()* pozvati dva puta. Nakon prvog poziva funkcije kao rezultat se vraća redak u kojem se nalazi najveći element. Ponovnim pozivom funkcije dolazi se do najvećeg elementa matrice. Normalizacija matrice omogućava skaliranje svih elemenata matrice na interval od nula do jedan gdje će vrijednost približno nula poprimiti elementi koji su značajno manji od najvećeg elementa, a vrijednost jedan elementi koji su približno jednaki ili jednaki najvećem elementu. Daljnjom selekcijom elemenata normirane matrice dolazimo do polja na kojima se nalaze figure. Za crna polja, u idealnom slučaju, suma svih piksela na mjestima gdje se ne nalaze figure trebala bi biti jednaka nula no budući da se uvijek mogu pojaviti smetnje potrebno je postaviti granicu tolerancije. U ovom slučaju koristi se vrijednost 0.03, odnosno ukoliko je vrijednost manja od 0.03 na tom polju se ne nalazi figura, u suprotnom se na tom polju nalazi figura. Istim postupkom, ali uz drugačije granice, određuju se i položaji figura na bijelim poljima. Nakon provedenog postupka formira se matrica figura na način da se zbroje matrice crnih i bijelih polja. Tako nastala matrica sastoji se od kombinacije nula i jedinica gdje polja na kojima se nalaze jedinice ujedno predstavljaju i polja na kojima se nalaze figure. Oduzimanjem matrice koja prati igru od matrice figura dolazimo do matrice koja ima oblik prikazan na slici 3.12.

matricaFigura

8x8 double

	1	2	3	4	5	6	7	8
1		0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	-1
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

Slika 3.12. *Prikaz matrice figura.*

Ukoliko dobro pogledamo izgled matrice figura može se vidjeti da u ovom slučaju postoje dvije vrijednosti koje se razlikuju od nule, odnosno vrijednosti minus jedan i jedan. Budući da se uspoređuju vrijednosti trenutnih položaja figura s položajima figura tijekom šahovske partije dolazi se do zaključka da je u ovom slučaju došlo do odigranog poteza, odnosno figura koja se nalazila na mjestu gdje je vrijednost minus jedan (koordinate polja (3,8)) sada se nalazi na mjestu gdje je vrijednost jedan (koordinate polja (6,5)). Sve što preostaje u ovom slučaju je ispisivanje odigranog poteza i ažuriranje svih relevantnih matrica vezanih uz šahovsku partiju. Nešto teži slučaj nastaje ukoliko je odigrani potez razmjena figura. U tom slučaju matrica figura poprima oblik prikazan na slici 3.13.

matricaFigura

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	-1	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

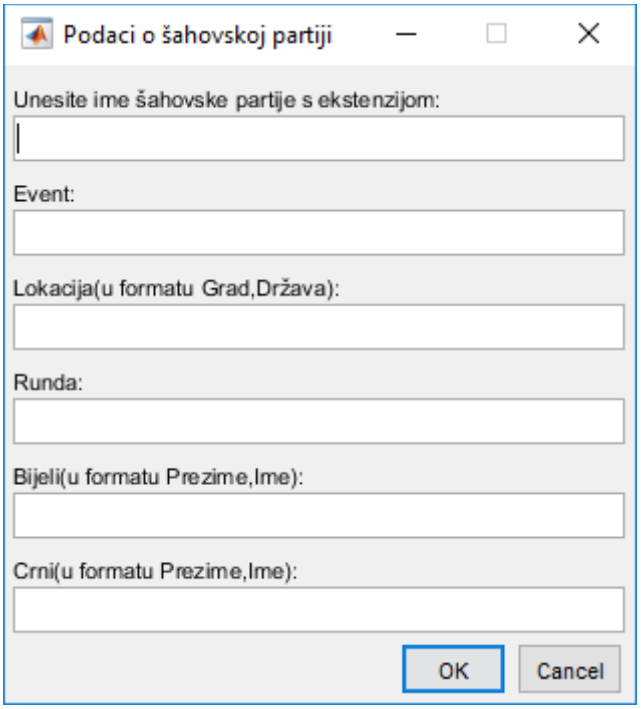
Slika 3.13. *Prikaz matrice figura.*

Za razliku od prethodnog slučaja gdje su postojale vrijednosti minus jedan i jedan, ovdje je slučaj nešto drugačiji. Sve što se na temelju ove matrice može zaključiti je da je figura prešla s polja koje ima koordinate (3,6). Kako bi otkrili na koje je polje navedena figura prešla, odnosno s kojom se figurom izvršila razmjena, potrebno je otkriti koja su polja doživjela najveću promjenu u odnosu na prethodni potez. Budući da je polje na kojem se dogodila najveća promjena upravo polje s kojeg je odigran potez, taj podatak se zanemaruje te se traži drugo polje po redu na kojem se dogodila najveća promjena jer se radi o polju na kojem je bijela figura zamijenjena crnom ili obrnuto. Nakon otkrivanja polja na kojem je došlo do razmijene figura ponovno preostaje ispisivanje odigranog poteza i ažuriranje relevantnih matrica.

Poseban slučaj odigranog poteza predstavlja šahovska rohada, odnosno zamjena položaja topa i kralja. U ovom slučaju do promijene dolazi na četiri polja u stacionarnom stanju u odnosu na polja prethodno odigranog poteza. U tom slučaju postojati će četiri polja na kojima će biti vrijednosti minus jedan i jedan u jednakom omjeru (2:2). Primjenom analognog postupka za detekciju odigranog poteza realizira se i ovaj poseban slučaj.

3.6. Ispis odigranih poteza

Pokretanjem glavne datoteke *main.m* (prilog A) inicira se početak izvođenja programa. Postupak detekcije odigranih poteza u matematičkom obliku prikazan je u prilogu B dok je u prilozima C, D, E, F i G prikazana realizacija ispisa odigranih poteza. Ispisivanje odigranih poteza omogućeno je na dva načina: ispis polja s kojeg je odigran potez i polja na koje je odigran potez te ispis u PGN formatu (vidi [7]). Prvi od navedenih je znatno jednostavniji način budući da je potrebno samo ispisati relevantna polja dok je PGN zapis znatno složeniji i zahtjeva stvaranje dodatnih matrica i razradu logike kako bi se mogao realizirati odgovarajući ispis odigranih šahovskih poteza. PGN zapis sastoji se od dva dijela. Prvi dio sadrži oznake koje uključuju ime događaja, lokaciju održavanja događaja, datum, broj runde, ime i prezime bijelog i crnog igrača, rezultat te dodatne oznake ovisno o potrebama. Drugi dio sadrži odigrane šahovske poteze u algebarskoj šahovskoj notaciji. Kako se na temelju videozapisa šahovske partije većina oznaka ne može dohvatiti, njihovo unošenje korisniku je omogućeno pri pokretanju programa kao što je i prikazano na slici 3.14.



Slika 3.14. *Unos podataka o šahovskoj partiji.*

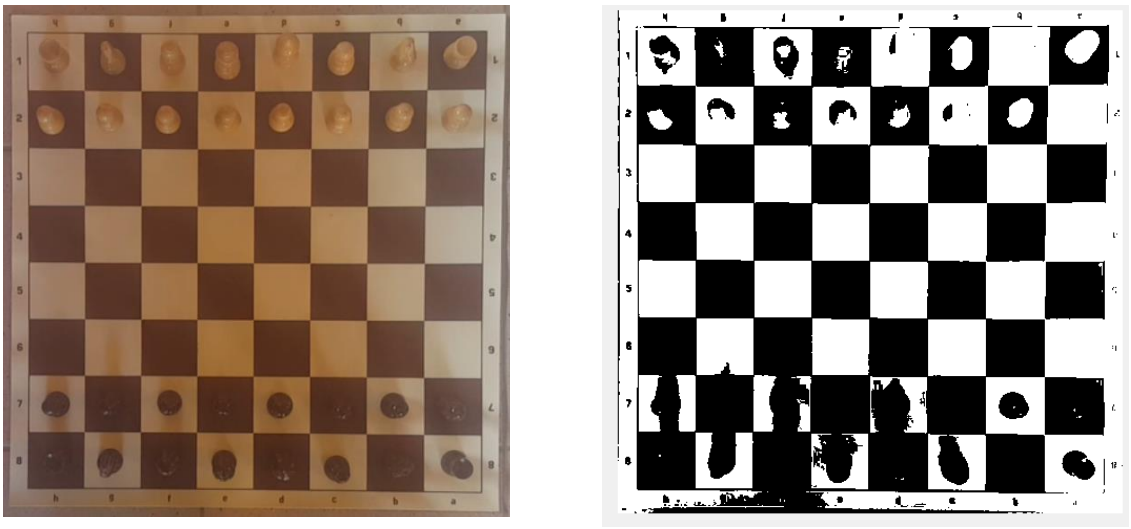
Kako bi se ostvario ispis korištenjem algebarske šahovske notacije uvedena je dodatna matrica koja sadrži oznake šahovskih figura ovisno o njihovom početnom položaju. Budući da je algoritam univerzalan, oznake figura automatski se podešavaju nakon prvog odigranog poteza te je samim time algoritam neovisan o orijentaciji videozapisa, odnosno o stranama na kojima se nalaze bijele figure i crne figure. Neovisnost o načinu postavljanja figura na šahovsku ploču omogućena je provjerom smjera kretanja bijelih figura nakon prvog odigranog poteza te zatim postavljanja oznaka polja kao i oznaka figura na temelju smjera. Iznimno je bitno pravilno postaviti notaciju polja budući da je ista striktno propisana (prvo polje u prvom redu i stupcu, najbliže bijeloj kraljici, ovisno o orijentaciji, uvijek ima oznaku a1). Oznake koje se koriste za označavanje figura su redom: P za pješaka, K za kralja, Q za kraljicu, R za topa, N za skakača i B za lovca. Prilikom ispisa odigranih poteza treba voditi računa o pravilima koja nalaže standardna algebarska šahovska notacija (SAN):

- numeriranje poteza vrši se pri svakom potezu koji odigra bijeli igrač,
- potez se ispisuje na način da se ispiše polje na koje je odigrana figura pomaknuta pri čemu se prije polja ispisuje oznaka figure, osim u slučaju pješaka (npr. e7, Be7),

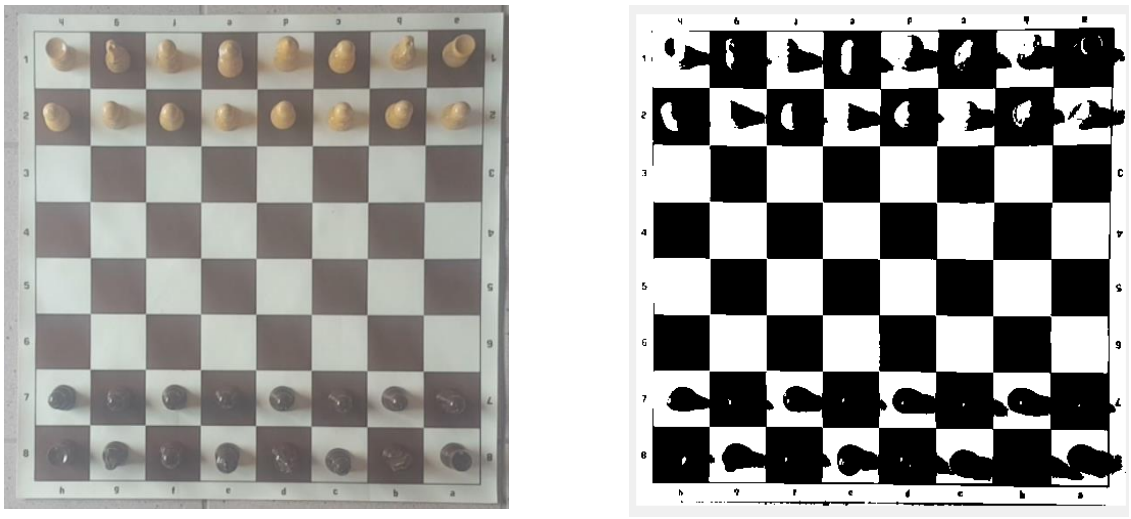
- ukoliko postoji mogućnost da dvije iste figure, primjerice dva skakača, mogu prijeći na isto polje potrebno je naglasiti o kojem se skakaču radi (npr. Nge2),
- ukoliko je u odigranom potezu došlo do promocije pješaka postavlja se znak '=' između oznake polja na koje je odigran pješak i figure u koju je promoviran (npr. e8=Q),
- ako je u odigranom potezu protivnička figura uklonjena s šahovske ploče, odnosno ukoliko je figura pojedena, koristi se znak 'x' , (npr. Qxf7+),
- ukoliko je odigran šah, na kraju notacije postavlja se znak '+' (npr. Qf7+),
- ako je odigrana rohada, tada se potez obilježava kao "O-O" s kraljeve strane, odnosno "O-O-O" s kraljičine strane,
- ukoliko je u odigranom potezu došlo do šah-mata, na kraju notacije postavlja se znak '#' (npr. Qf7#).

4. EKSPERIMENTALNI REZULTATI

Rezultati dobiveni primjenom prethodno opisane metode direktno ovise o uvjetima u kojima je videozapis šahovske partije stvoren. Poželjno je da se snimanje videozapisa provodi u kontroliranim uvjetima, gdje je iznimno dobro osvjetljenje kako bi se otklonio utjecaj sjene. Za potrebe testiranja metode kreirana su tri videozapisa, svaki u drugačijim uvjetima i drugačije razlučivosti. Videozapisi korišteni u nastavku snimljeni su s rezolucijom $1280x720$ i 30 slika u sekundi. Na slikama 3.15. i 3.16. možemo vidjeti utjecaj sjene na prepoznavanje odigranih poteza. Na temelju slike u binarnom obliku u iznimno lošim svjetlosnim uvjetima gotovo je nemoguće prepoznati pojedine figure dok neke nisu niti djelomično vidljive. Također se može vidjeti velik utjecaj smetnji u preostalim poljima prvenstveno uzrokovane stvaranjem sjene.

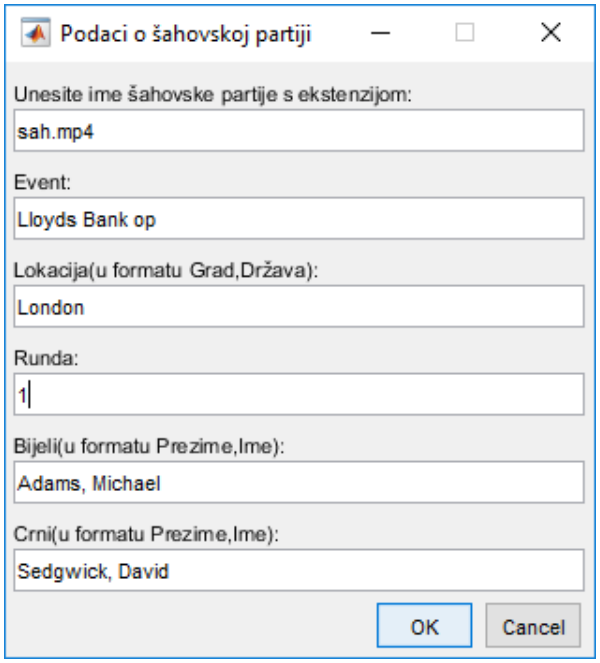


Slika 3.15. *Isječak iz videozapisa (lijevo) i binarna slika (desno).*

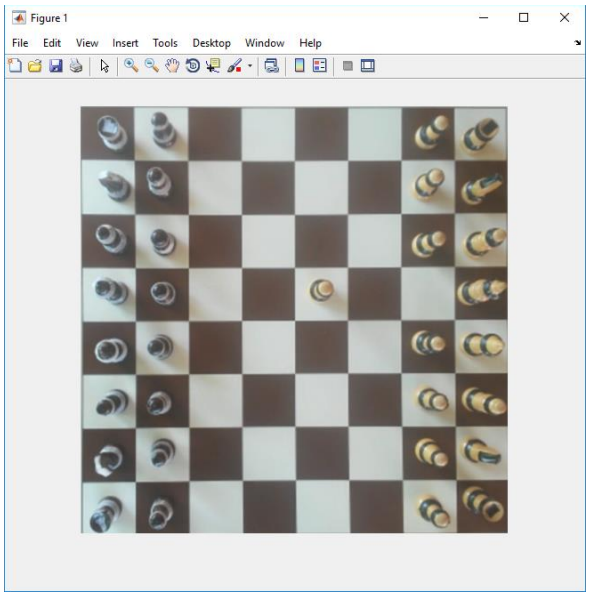


Slika 3.16. *Isječak iz videozapisa (lijevo) i binarna slika (desno).*

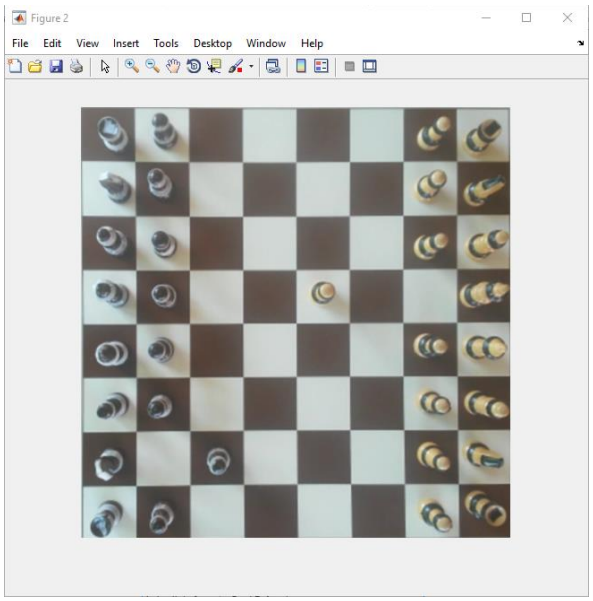
Na slikama 3.17. do 3.26. u nastavku prikazan je primjer rada programa na videozapisu snimljenom s dobrim svjetlosnim uvjetima s prethodno spomenutim postavkama u potpoglavlju 3.1. Slika 3.17. prikazuje unesene proizvoljne informacije o odigranoj šahovskoj partiji dok slike 3.18. do 3.26. prikazuju slike videozapisa na kojima su detektirani odigrani potezi i njihov ispis u MATLAB programskom okruženju i datoteci u PGN formatu stvorenoj na temelju unesenih podataka o odigranoj šahovskoj partiji.



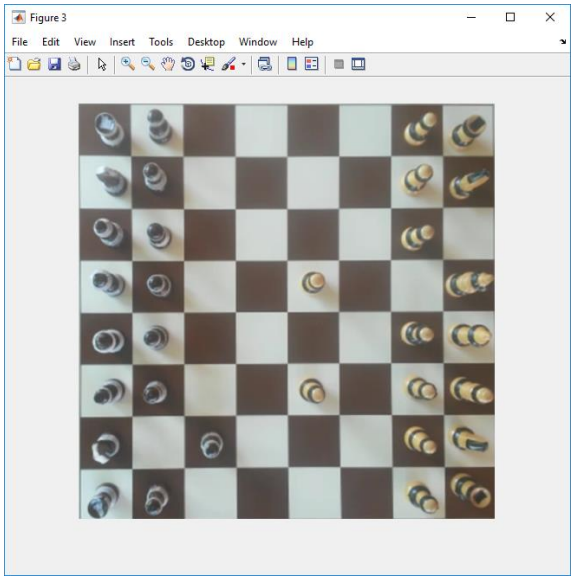
Slika 3.17. *Unos podataka o odigranoj šahovskoj partiji.*



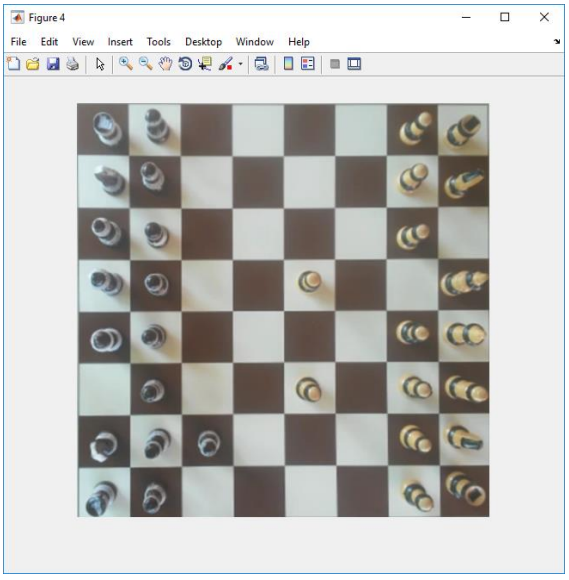
Slika 3.18. *Prvi odigrani potez.*



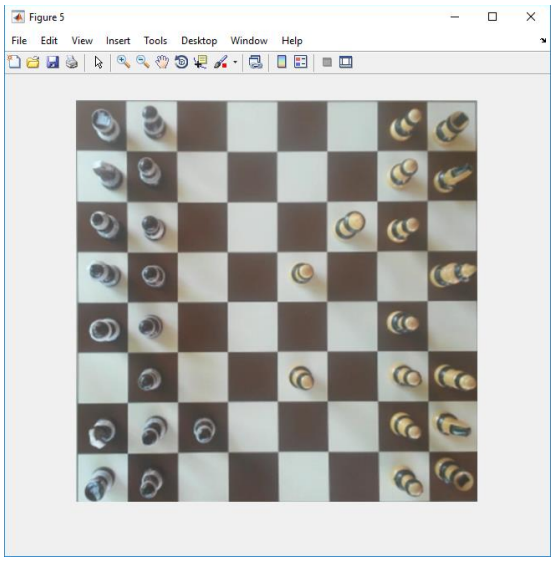
Slika 3.19. *Drugi odigrani potez.*



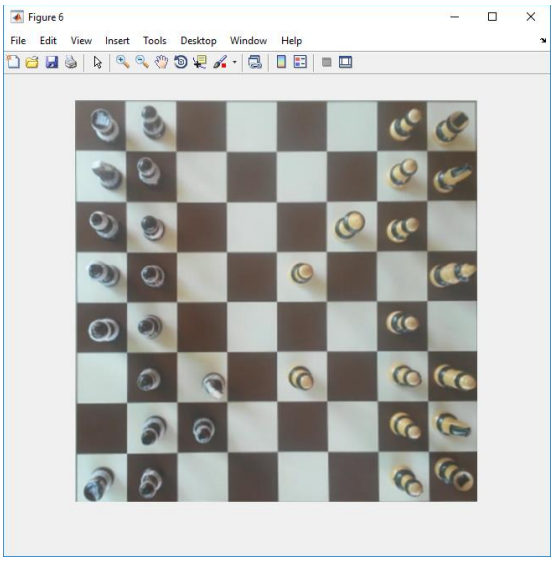
Slika 3.20. *Treći odigrani potez.*



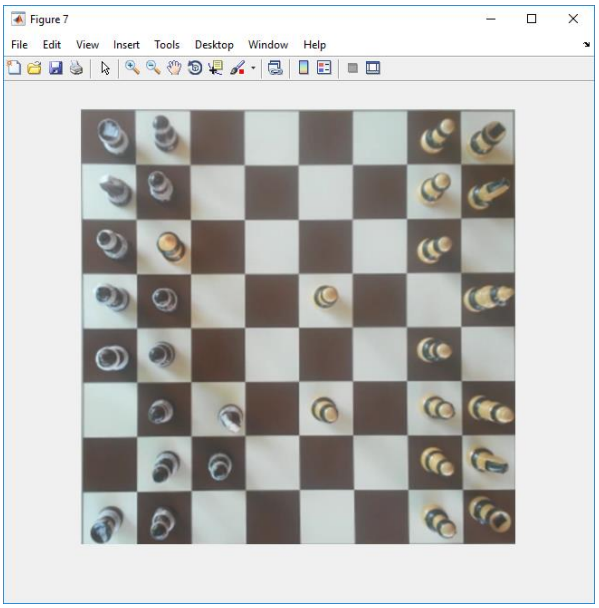
Slika 3.21. *Četvrti odigrani potez.*



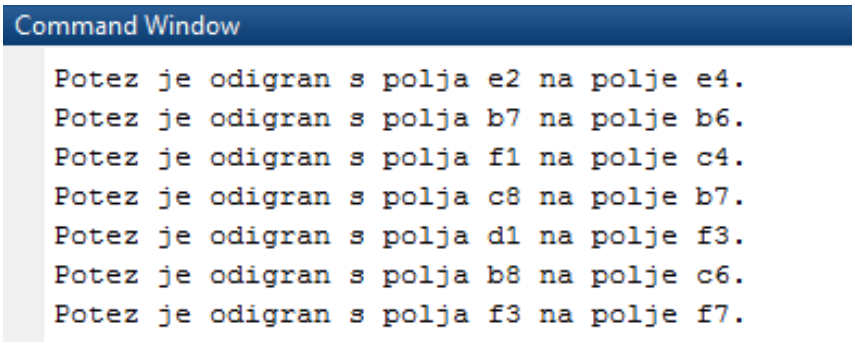
Slika 3.22. *Peti odigrani potez.*



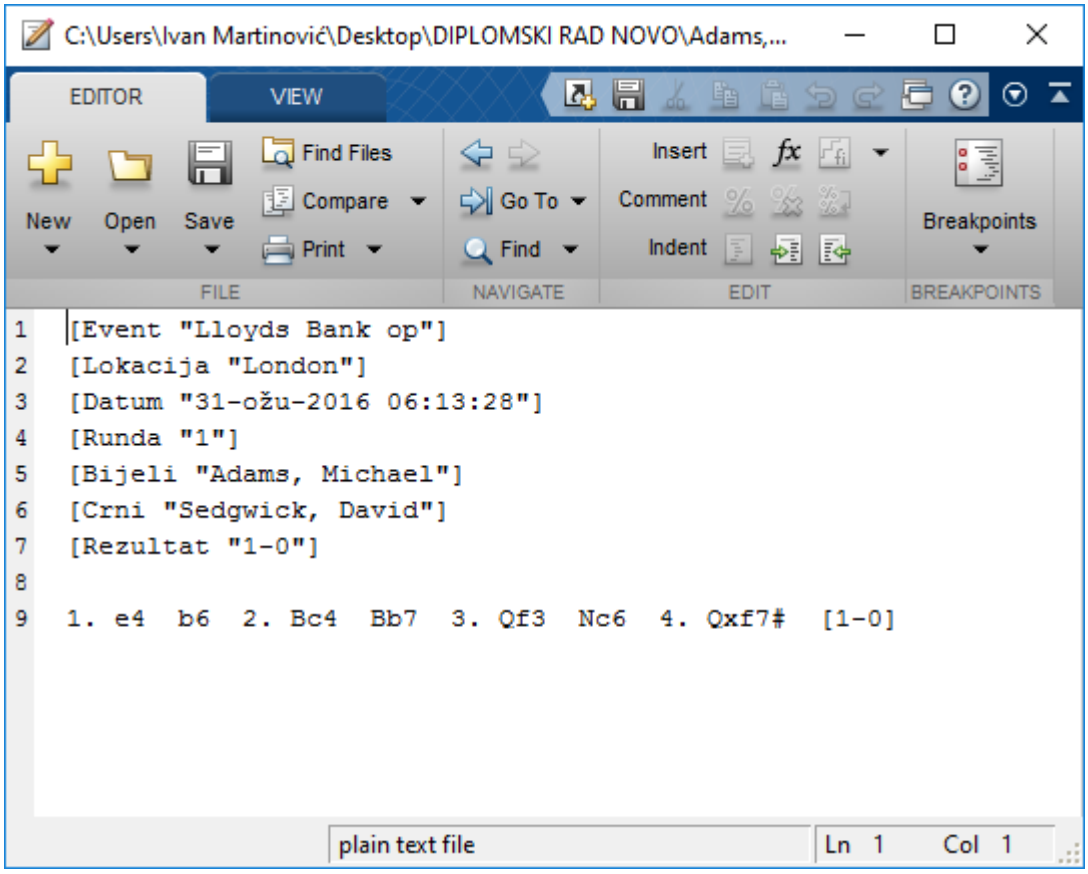
Slika 3.23. *Šesti odigrani potez.*



Slika 3.24. *Sedmi odigrani potez.*



Slika 3.25. *Slika detektiranih poteza za primjere na slikama 3.16. – 3.22.*



Slika 3.26. *Ispis odigranih poteza u PGN formatu.*

5. ZAKLJUČAK

U ovom radu izrađen je program koji detektira i ispisuje odigrane poteze na temelju video snimke odigrane šahovske partije. Cjelokupni sustav se može podijeliti u četiri glavne cjeline: priprema i učitavanje video snimke šahovske partije, priprema slike za obradu, obrada slike i obrada podataka te detekcija i ispis odigranih šahovskih poteza. Program je uspješno realiziran te radi prema očekivanju. Prilikom realizacije programa izvršena su brojna pojednostavljenja s ciljem smanjenja složenosti cjelokupnog problema, od načina snimanja šahovske partije pa sve do načina obrade slike i ispisa odigranih poteza. Prednosti korištenja ovog rješenja i korištenih metoda omogućavaju iznimno brzu i efikasnu obradu podataka (prosječno se obradi i do 80 slika u sekundi) uz više nego dobre rezultate. Glavna mana programa jest ovisnost o uvjetima u kojima je video snimka stvorena, odnosno uspješnost znatno ovisi prvenstveno o svjetlosnim uvjetima, ali i o odnosu boja šahovske ploče i šahovskih figura.

LITERATURA

[1] http://goo.gl/zV5cde , ožujak 2016.

[2] T. Andrews, Computation Time Comparison Between Matlab and C++ Using Launch Windows, California Polytechnic State University San Luis Obispo, SAD, rujan 2016.

[3] http://goo.gl/2VLraK , ožujak 2016.

[4] http://goo.gl/EAy18r , ožujak 2016.

[5] http://goo.gl/SDgp5r , ožujak 2016.

[6] P. Viola, M. J. Jones, Robust Real-Time Face Detection, International Journal Of Computer Vision 57(2), 137 – 154, 2004., rujan 2016.

[7] https://goo.gl/21kTk2, rujan 2016.

SAŽETAK

U ovom radu izrađen je program koji detektira i ispisuje odigrane poteze na temelju video snimke odigrane šahovske partije. Cjelokupni sustav se može podijeliti u četiri glavne cjeline: priprema i učitavanje video snimke šahovske partije, priprema slike za obradu, obrada slike i obrada podataka te detekcija i ispis odigranih šahovskih poteza. Kroz rad razmotrene su brojne metode obrade slike, mogućnosti primjene pojedinih metoda, njihove prednosti i nedostaci kao i razlozi korištenja navedenih metoda. Nakon opisa problema i korištenih alata, detaljno je opisana realizacija programa te su navedene alternativne metode realizacije problemskog zadatka koje pružaju slične ili jednake rezultate.

Ključne riječi: MATLAB, šah, obrada slike, histogram, obrada podataka.

ABSTRACT

In this thesis chess move recognition and printing program was developed. It consists of four main subsystems: preparation and loading of a video of a chess game, video preparation for image processing, image processing and data processing, chess move detection and printing. First, various methods of image processing, their application possibilites and their advantages and disadvantages are described along with task explanation and used tools. After that, system implementation is explained in great detail along with the results. At the end of thesis, alternative methods for solving the problem are given and explained.

Keywords: MATLAB, chess, image processing, histogram, data processing.

ŽIVOTOPIS

Ivan Martinović rođen je 25. listopada 1990. godine u Vukovaru. Od svoga rođenja živi u Iloku gdje pohađa osnovnu školu. Sudjelovao je na natjecanjima iz matematike i informatike. Opću gimnaziju u Iloku završava 2009. godine, te iste godine upisuje preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku kojega redovno završava školske godine 2012/13. Iste godine upisuje diplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. Od 2015. godine uz redoviti studij radi kao profesor fizike i elektrotehničke grupe predmeta u Srednjoj školi Ilok te kao *freelancer* programer u struci.

PRILOZI

Kompletan izvorni kod realiziranog programa, kao i svi dodatni materijali (video snimke, slike itd.) nalaze se na DVD-u priloženom uz ovaj rad. U nastavku slijedi ispis izvornog koda.

Prilog A. *main.m*

```
1     PrepoznavanjeOdigranihPoteza

2     brojOdigranihPoteza = 0;

3     s = regexprep(s,'+','#');
4     matricaPoteza(temp4-1,1) = cell({s});
5     for i = 1:size(matricaPoteza)
6         if cell2mat(matricaPoteza(i,1)) ~= 0
7             brojOdigranihPoteza = brojOdigranihPoteza + 1;
8         end
9     end
10    if mod(brojOdigranihPoteza,2) ~= 0 && s(size(s,2)) == '#'
11        rezultat = [1 0];
12    elseif mod(brojOdigranihPoteza,2) == 0 && s(size(s,2)) == '#'
13        rezultat = [0 1];
14    else
15        rezultat = [0 0];
16    end
17    datotekaInfo = dir(char(datoteka(1,1)));
18    vrijeme = datotekaInfo.date;
19    imeDatoteke = regexprep(strcat(char(datoteka(5,1)),'-',char(datoteka(6,1))),'
','');
20    dat = fopen(imeDatoteke,'w');
21        fprintf(dat,'[Event "%s"] \n', char(datoteka(2,1)));
22        fprintf(dat,'[Lokacija "%s"] \n', char(datoteka(3,1)));
23        fprintf(dat,'[Datum "%s"] \n',vrijeme);
24        fprintf(dat,'[Runda "%s"] \n', char(datoteka(4,1)));
25        fprintf(dat,'[Bijeli "%s"] \n', char(datoteka(5,1)));
26        fprintf(dat,'[Crni "%s"] \n', char(datoteka(6,1)));
27        if rezultat(1,1) == 1 || rezultat(1,2) == 1
28            fprintf(dat,'[Rezultat "%d-%d"] \n\n',rezultat(1,1),rezultat(1,2));
29        else
30            fprintf(dat,'[Rezultat "1/2-1/2"] \n\n');
31        end
32        for i=1:brojOdigranihPoteza;
33            if mod(i,2) ~= 0
34                fprintf(dat,'%d. ',floor(i/2)+1);
35            end
36            fprintf(dat,'%s  ',cell2mat(matricaPoteza(i,1)));
37            if mod(i,7) == 0
38                fprintf('\n');
39            end
```

```
40            if i == brojOdigranihPoteza
41                if rezultat(1,1) == 1 || rezultat(1,2) == 1
42                    fprintf(dat,'[%d-%d]', rezultat);
43                else
44                    fprintf(dat,'[1/2-1/2]');
45                end
46            end
47        end
48    fclose(dat);
```

Prilog B. *PrepoznavanjeOdigranihPoteza.m*

```
1     clear; close all; clc; warning('off','all');
2     datoteka = inputdlg({'Unesite ime šahovske partije s ekstenzijom: ',...
3         'Event:  ','Lokacija(u formatu Grad,Država): ','Runda: '...
4         ,'Bijeli(u formatu Prezime,Ime): ',...
5         'Crni(u formatu Prezime,Ime): '},'Podaci o šahovskoj partiji',[1 50]);
6     video = VideoReader(char(datoteka(1,1)));
7     imshow(read(video,10));
8     msgbox('Označite vrhove rubnih šahovskih polja!','Value','tex','modal');
9     [y,x] = ginput(4);
10    x1 = [ 0 0 0 0 ];
11    y1 = [ 0 0 0 0 ];
12    for i=1:4 %sortiranje koordinata odabranih rubnih točaka
13        if ( abs(y(i) - min(y)) < 0.1 * y(i) && abs(x(i)- min(x)) < 0.1 * x(i) )
14            x1(1) = x(i);
15            y1(1) = y(i);
16        elseif ( abs(y(i) - max(y)) < 0.1 * y(i) && abs(x(i) - max(x)) < 0.1 * x(i)
)
17            x1(4) = x(i);
18            y1(4) = y(i);
19        else
20            if ( x1(2)  == 0 && y1(2) == 0 )
21                x1(2) = x(i);
22                y1(2) = y(i);
23            else
24                if ( y(i) > y1(2))
25                    y1(3) = y1(2);
26                    x1(3) = x1(2);
27                    y1(2) = y(i);
28                    x1(2) = x(i);
29                else
30                    y1(3) = y(i);
31                    x1(3) = x(i);
32                end
33            end
34        end
35    end% END - sortiranje
36    close Figure 1;
37    x = x1; y = y1;
```

```
redniBrojPoteza = 0;
odigraniPotez = [ 0 0 ];
odigraniPotez2 = [ 0 0 ];
smjerKretanjaBijelihFigura = 0;% 0 ->, 1 <-, 2 - od gore prema dolje,
3- od dolje prema gore
potez = 0;
polja = double(zeros(8,8));
poljaBijela = double(zeros(8,8));
poljaCrna = double(zeros(8,8));
prvaSlika = 1;
prviPotez = 1;
BrojSlika = floor(video.Duration * video.FrameRate) - 5;
prethodnaPolja = double(zeros(8,8));
promjenaPolja = 0;
temp4 = 1;
matricaPoteza = cell({zeros(120,1)});
poljaPrethodnoOtkrivenogPoteza = double(zeros(8,8));
h = waitbar(0,'Obrada je u tijeku...');
for brojSlike = 1:BrojSlika
    waitbar (brojSlike/BrojSlika)
    Slika = read(video,brojSlike);
    SlikaResize = imresize(Slika(x(1): x(4), y(1):y(4),:), [480 480]);
    %imshow(SlikaResize);
    r = SlikaResize(:,:,1);
    b = SlikaResize(:,:,2);
    g = SlikaResize(:,:,3);
    SlikaResize(:,:,1) = histeq(r);
    SlikaResize(:,:,2) = histeq(b);
    SlikaResize(:,:,3) = histeq(g);
    SlikaBinarna = im2bw(SlikaResize,graythresh(SlikaResize));
    integralnaSlika = cumsum(cumsum(double(SlikaBinarna),2));%->integralna slika
    polja(1,1) = integralnaSlika(56,56)-integralnaSlika(4,56) -
    integralnaSlika(56,4) + integralnaSlika(4,4); % razvrstavanje suma polja
    for i = 2:8 %prvi stupac i prvi redak šahovske ploče
    polja(1,i) = integralnaSlika(56,(i*60)-4) - integralnaSlika(56,((i-1)*60)+4)
     - integralnaSlika(4,(i*60)-4) + integralnaSlika(4,((i-1)*60)+4);
    polja(i,1) = integralnaSlika((i*60)-4,56) - integralnaSlika(((i-1)*60)+4,56)
     - integralnaSlika((i*60)-4,4) + integralnaSlika(((i-1)*60)+4,4);
    end
    for i = 2:1:8 %ostatak polja
        for  j = 2:1:8
            polja(i,j) = integralnaSlika((i*60)-4,(116+((j-2)*60))) -
            integralnaSlika((i*60)-4,(64+((j-2)*60))) -
            integralnaSlika((((i-1)*60)+4),((56+(j-1)*60))) +
            integralnaSlika(((i-1)*60)+4,((64+((j-2)*60))));
        end
    end
    if (prvaSlika == 1)% određivanje početnih položaja figura i  oznaka polja
        if ( polja (3,2) > polja(4,2) )
            matricaIgre = [ 1 1 0 0 0 0 1 1;1 1 0 0 0 0 1 1;
                            1 1 0 0 0 0 1 1;1 1 0 0 0 0 1 1;
                            1 1 0 0 0 0 1 1;1 1 0 0 0 0 1 1;
```

```
                                1 1 0 0 0 0 1 1;1 1 0 0 0 0 1 1];
            OznakeFigura = cell({'RW','PW','-','-','-','-','PB','RB';
                                 'NW','PW','-','-','-','-','PB','NB';
                                 'BW','PW','-','-','-','-','PB','BB';
                                 'QW','PW','-','-','-','-','PB','QB';
                                 'KW','PW','-','-','-','-','PB','KB';
                                 'BW','PW','-','-','-','-','PB','BB';
                                 'NW','PW','-','-','-','-','PB','NB';
                                 'RW','PW','-','-','-','-','PB','RB'});
            polozajFigura = cell({'a1','a2','a3','a4','a5','a6','a7','a8';
                                  'b1','b2','b3','b4','b5','b6','b7','b8';
                                  'c1','c2','c3','c4','c5','c6','c7','c8';
                                  'd1','d2','d3','d4','d5','d6','d7','d8';
                                  'e1','e2','e3','e4','e5','e6','e7','e8';
                                  'f1','f2','f3','f4','f5','f6','f7','f8';
                                  'g1','g2','g3','g4','g5','g6','g7','g8';
                                  'h1','h2','h3','h4','h5','h6','h7','h8'});
        else
            matricaIgre = [ 1 1 1 1 1 1 1 1;1 1 1 1 1 1 1 1;
                            0 0 0 0 0 0 0 0;0 0 0 0 0 0 0 0;
                            0 0 0 0 0 0 0 0;0 0 0 0 0 0 0 0;
                            1 1 1 1 1 1 1 1;1 1 1 1 1 1 1 1];
            OznakeFigura = cell({'RB','NB','BB','QB','KB','BB','NB','RB';
                                 'PB','PB','PB','PB','PB','PB','PB','PB';
                                 '-','-','-','-','-','-','-','-';
                                 '-','-','-','-','-','-','-','-';
                                 '-','-','-','-','-','-','-','-';
                                 '-','-','-','-','-','-','-','-';
                                 'PW','PW','PW','PW','PW','PW','PW','PW';
                                 'RW','NW','BW','QW','KW','BW','NW','RW'});
            polozajFigura =cell({'a8','b8','c8','d8','e8','f8','g8','h8';
                                 'a7','b7','c7','d7','e7','f7','g7','h7';
                                 'a6','b6','c6','d6','e6','f6','g6','h6';
                                 'a5','b5','c5','d5','e5','f5','g5','h5';
                                 'a4','b4','c4','d4','e4','f4','g4','h4';
                                 'a3','b3','c3','d3','e3','f3','g3','h3';
                                 'a2','b2','c2','d2','e2','f2','g2','h2';
                                 'a1','b1','c1','d1','e1','f1','g1','h1'});
            smjerKretanjaBijelihFigura = 3;
        end
    end
    for i = 1:1:8
        for j=1:1:8
            if ( polja(i,j) >= 0.99 * prethodnaPolja(i,j) && polja(i,j) <=
            1.01 * prethodnaPolja(i,j))
            else
                promjenaPolja = promjenaPolja + 1;
            end
        end
    end
    if (promjenaPolja == 2 || promjenaPolja == 1 || promjenaPolja == 4)
        if polja(4,4) < polja(5,4) && prvaSlika == 1
```

```
            temp7 = 1;
            prvaSlika = 0;
        elseif polja(4,4) > polja(5,4) && prvaSlika == 1
            temp7 = 0;
            prvaSlika = 0;
        end
        if(temp7 == 1) %podijela na bijela i crna polja
            for i = 1:1:8
                for j=1:2:8
                    if(mod(i,2)~= 0)
                        poljaCrna(i,j) = polja(i,j);
                        poljaBijela(i,j+1) = polja(i,j+1);
                    else
                        poljaCrna(i,j+1) = polja(i,j+1);
                        poljaBijela(i,j) = polja(i,j);
                    end
                end
            end
        else
            for i = 1:1:8
                for j=1:2:8
                    if(mod(i,2)~= 0)
                        poljaBijela(i,j) = polja(i,j);
                        poljaCrna(i,j+1) = polja (i,j+1);
                    else
                        poljaBijela(i,j+1) = polja(i,j+1);
                        poljaCrna(i,j) = polja (i,j);
                    end
                end
            end
        end
        poljaCrna = poljaCrna/max(max(poljaCrna));
        poljaBijela = poljaBijela/max(max(poljaBijela));
        for i = 1:1:8
            for j=1:1:8
                if ( poljaCrna(i,j) < 0.03 )
                    poljaCrna(i,j) = 0;
                else
                    poljaCrna(i,j) = 1;
                end
                if (poljaBijela(i,j) > 0.97 || poljaBijela(i,j) == 0 )
                    poljaBijela(i,j) = 0;
                else
                    poljaBijela(i,j) = 1;
                end
            end
        end
        matricaFigura = poljaCrna + poljaBijela;
        matricaFigura = matricaFigura - matricaIgre;
        for i = 1:1:8
            for j=1:1:8
                if (matricaFigura(i,j) == 1)
```

```
186                     temp1 = temp1 + 1;
187                     if temp1 > 1
188                         odigraniPotez3 = [i,j];
189                     else
190                         odigraniPotez = [i,j];
191                     end
192                 elseif (matricaFigura(i,j) == -1)
193                         temp2 = temp2 + 1;
194                         if temp2 > 1
195                             odigraniPotez4 = [i,j];
196                         else
197                             odigraniPotez2 = [i,j];
198                         end
199                 elseif (promjenaPolja == 1 && temp3 == 0)
                    %provjera je li došlo do zamjene figura
200                     [maksimum,indeks] =max(abs(poljaPrethodnoOtkrivenogPoteza(:)
                    - polja(:)));
201                     [m,n] = ind2sub(size(poljaPrethodnoOtkrivenogPoteza)
                    ,indeks);
202                     if( m == odigraniPotez2(1,1) && n == odigraniPotez2(1,2))
203                         poljaPrethodnoOtkrivenogPoteza(m,n) = polja(m,n);
204                         [maksimum,indeks] = max(abs(
                         poljaPrethodnoOtkrivenogPoteza(:) - polja(:)));
205                         [m,n] = ind2sub(size(poljaPrethodnoOtkrivenogPoteza)
                         ,indeks);
206                         odigraniPotez = [m,n];
207                         temp1 = temp1 + 1;
208                         temp3 = temp3 + 1;
209                     end
210                 elseif (promjenaPolja == 4 && temp1 == 2 && temp2 == 2)
                    %odigrana rohada
211                     provjeraRohade
212                 end
213             end
214         end
215         if ( temp1 == 1 && temp2 == 1)%pozivanje ispisa odigranog poteza
216             ispisPoteza
217             matricaIgre(odigraniPotez2(1,1),odigraniPotez2(1,2)) = 0;
218             matricaIgre(odigraniPotez(1,1),odigraniPotez(1,2)) = 1;
219             poljaPrethodnoOtkrivenogPoteza = polja;
220         end
221     end
222     prethodnaPolja = polja;
223     promjenaPolja = 0;
224     temp1 = 0;
225     temp2 = 0;
226     temp3 = 0;
227     bijeli = 0; crni = 0;
228 end
229 delete(h)
```

Prilog C. *ispisPoteza.m*

```
1     if prviPotez == 1 %podešavanje notacije ovisno o smjeru kretanja bijelih figura
2         if matricaIgre(3,2) == 1
3             if odigraniPotez(1,1) == 1||2||3||4||5||6||7||8
                && odigraniPotez(1,2) == 5||6
4                 OznakeFigura = rot90(OznakeFigura,2);
5                 polozajFigura = rot90(polozajFigura,2);
6                 smjerKretanjaBijelihFigura = 1;
7             end
8         else
9             if odigraniPotez(1,1) == 3||4 &&
                odigraniPotez(1,2) == 1||2||3||4||5||6||7||8
10                OznakeFigura = rot90(OznakeFigura,2);
11                polozajFigura = rot90(polozajFigura,2);
12                smjerKretanjaBijelihFigura = 2;
13            end
14        end
15        prviPotez = 0;
16    end
17    temp5 = 1;
18    potez = potez + 1;
19    s1 = cell2mat(polozajFigura(odigraniPotez(1,1),odigraniPotez(1,2)));
20    s2 = cell2mat(polozajFigura(odigraniPotez2(1,1),odigraniPotez2(1,2)));
21    odigranaFigura = cell2mat(OznakeFigura(odigraniPotez2(1,1),
      odigraniPotez2(1,2)));
22    prethodnoPoljeOdigraneFigure = cell2mat(OznakeFigura(odigraniPotez(1,1),
      odigraniPotez(1,2)));
23    if mod(potez,2)~= 0 %ako je br. poteza neparan -> bijeli na potezu
24        bijeli = 1;
25        s3 = 'KB';
26    else %ako je br. poteza paran -> crni na potezu
27        crni = 1;
28        s3 = 'KW';
29    end
30    temp6 = odigranaFigura;
31    odigranaFigura = odigranaFigura(1);
32    OznakeFigura{odigraniPotez2(1,1),odigraniPotez2(1,2)} = '-';
33    OznakeFigura{odigraniPotez(1,1),odigraniPotez(1,2)} = temp6;
34    provjeraFigura
35    matricaPoteza(temp4,1) = cell({s});
36    temp4 =  temp4 + 1;
37    fprintf('Potez je odigran s polja %s na polje %s. \n',s2,s1)
```

Prilog D. *provjeraFigura.m*

```
1     s = '';
2     if odigranaFigura == 'P'% provjera poteza pijuna - promocija i mogućnost šaha
3         s = strcat(s,s1);
4         if smjerKretanjaBijelihFigura == 1
5             if (odigraniPotez(1,2) == 1 && s3(2) == 'W')||
```

```
            (odigraniPotez(1,2) == 8 && s3(2) == 'B')
6               s = strcat(s,'=Q');
7               if s3(2) == 'W'
8                   OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QW'});
9               else
10                  OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QB'});
11              end
12          end
13      if bijeli == 1
14          if odigraniPotez(1,1) == 1
15              if odigraniPotez(1,2) == 1
16              else
17                  if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) - 1),s3) == 1
18                      s = strcat(s,'+');
19                  end
20              end
21          elseif odigraniPotez(1,1) == 8
22              if odigraniPotez(1,2) == 1
23              else
24                  if strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                    odigraniPotez(1,2) - 1),s3) == 1
25                      s = strcat(s,'+');
26                  end
27              end
28          else
29              if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                odigraniPotez(1,2) - 1),s3) == 1 || ...
30              strcmp(OznakeFigura(odigraniPotez(1,1) -
                1,odigraniPotez(1,2) - 1),s3) == 1
31                  s = strcat(s,'+');
32              end
33          end
34      else
35          if odigraniPotez(1,1) == 1
36              if odigraniPotez(1,2) == 8
37              else
38                  if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) + 1),s3) == 1
39                      s = strcat(s,'+');
40                  end
41              end
42          elseif odigraniPotez(1,1) == 8
43              if odigraniPotez(1,2) == 8
44              else
45                  if strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                     odigraniPotez(1,2) + 1),s3) == 1
46                      s = strcat(s,'+');
47                  end
48              end
```

```
49            else
50                if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                  odigraniPotez(1,2) + 1),s3) == 1 || ...
51                 strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                   odigraniPotez(1,2) + 1),s3) == 1
52                    s = strcat(s,'+');
53                end
54            end
55        end
56        elseif smjerKretanjaBijelihFigura == 0
57            if (odigraniPotez(1,2) == 8 && s3(2) == 'W')||
                (odigraniPotez(1,2) == 1 && s3(2) == 'B')
58                s = strcat(s,'=Q');
59                if s3(2) == 'W'
60                    OznakeFigura(odigraniPotez(1,1),
                      odigraniPotez(1,2)) = cell({'QW'});
61                else
62                    OznakeFigura(odigraniPotez(1,1),
                      odigraniPotez(1,2)) = cell({'QB'});
63                end
64            end
65            if bijeli == 1
66                if odigraniPotez(1,1) == 1
67                    if odigraniPotez(1,2) == 8
68                    else
69                        if strcmp(OznakeFigura(odigraniPotez(1,1) +
                          1,odigraniPotez(1,2) + 1),s3) == 1
70                            s = strcat(s,'+');
71                        end
72                    end
73                elseif odigraniPotez(1,1) == 8
74                    if odigraniPotez(1,2) == 8
75                    else
76                        if strcmp(OznakeFigura(odigraniPotez(1,1) -
                          1,odigraniPotez(1,2) + 1),s3) == 1
77                            s = strcat(s,'+');
78                        end
79                    end
80                else
81                    if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                      odigraniPotez(1,2) + 1),s3) == 1 || ...
82                    strcmp(OznakeFigura(odigraniPotez(1,1) -
                      1,odigraniPotez(1,2) + 1),s3) == 1
83                        s = strcat(s,'+');
84                    end
85                end
86            else
87                if odigraniPotez(1,1) == 1
88                    if odigraniPotez(1,2) == 1
89                    else
90                        if strcmp(OznakeFigura(odigraniPotez(1,1) +
                          1,odigraniPotez(1,2) - 1),s3) == 1
```

```
91                          s = strcat(s,'+');
92                      end
93                  end
94              elseif odigraniPotez(1,1) == 8
95                  if odigraniPotez(1,2) == 1
96                  else
97                      if strcmp(OznakeFigura(odigraniPotez(1,1) -
                        1,odigraniPotez(1,2) - 1),s3) == 1
98                          s = strcat(s,'+');
99                      end
100                 end
101             else
102                 if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) - 1),s3) == 1 || ...
103                 strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                     odigraniPotez(1,2) - 1),s3) == 1
104                     s = strcat(s,'+');
105                 end
106             end
107         end
108     elseif smjerKretanjaBijelihFigura == 2
109         if (odigraniPotez(1,1) == 8 && s3(2) == 'W')||
             (odigraniPotez(1,1) == 1 && s3(2) == 'B')
110             s = strcat(s,'=Q');
111             if s3(2) == 'W'
112                 OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QW'});
113             else
114                 OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QB'});
115             end
116         end
117         if bijeli == 1
118             if odigraniPotez(1,2) == 1
119                 if odigraniPotez(1,1) == 8
120                 else
121                     if strcmp(OznakeFigura(odigraniPotez(1,1) +
                         1,odigraniPotez(1,2) + 1),s3) == 1
122                         s = strcat(s,'+');
123                     end
124                 end
125             elseif odigraniPotez(1,2) == 8
126                 if odigraniPotez(1,1) == 8
127                 else
128                     if strcmp(OznakeFigura(odigraniPotez(1,1) +
                         1,odigraniPotez(1,2) - 1),s3) == 1
129                         s = strcat(s,'+');
130                     end
131                 end
132             else
133                 if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) - 1),s3) == 1 || ...
```

```
134                         strcmp(OznakeFigura(odigraniPotez(1,1) +
                            1,odigraniPotez(1,2) + 1),s3) == 1
135                     s = strcat(s,'+');
136                 end
137             end
138         else
139             if odigraniPotez(1,2) == 1
140                 if odigraniPotez(1,1) == 1
141                 else
142                     if strcmp(OznakeFigura(odigraniPotez(1,1) -
                        1,odigraniPotez(1,2) + 1),s3) == 1
143                         s = strcat(s,'+');
144                     end
145                 end
146             elseif odigraniPotez(1,2) == 8
147                 if odigraniPotez(1,1) == 1
148                 else
149                     if strcmp(OznakeFigura(odigraniPotez(1,1) -
                        1,odigraniPotez(1,2) - 1),s3) == 1
150                         s = strcat(s,'+');
151                     end
152                 end
153             else
154                 if strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                    odigraniPotez(1,2) - 1),s3) == 1 || ...
155                 strcmp(OznakeFigura(odigraniPotez(1,1) -
                    1,odigraniPotez(1,2) + 1),s3) == 1
156                     s = strcat(s,'+');
157                 end
158             end
159         end
160     elseif smjerKretanjaBijelihFigura == 3
161         if (odigraniPotez(1,1) == 1 && s3(2) == 'W')||
           (odigraniPotez(1,1) == 8 && s3(2) == 'B')
162             s = strcat(s,'=Q');
163             if s3(2) == 'W'
164                 OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QW'});
165             else
166                 OznakeFigura(odigraniPotez(1,1),
                    odigraniPotez(1,2)) = cell({'QB'});
167             end
168         end
169         if bijeli == 1
170             if odigraniPotez(1,2) == 1
171                 if odigraniPotez(1,1) == 1
172                 else
173                     if strcmp(OznakeFigura(odigraniPotez(1,1) -
                        1,odigraniPotez(1,2) + 1),s3) == 1
174                         s = strcat(s,'+');
175                     end
176                 end
```

```
177             elseif odigraniPotez(1,2) == 8
178                 if odigraniPotez(1,1) == 1
179                     if strcmp(OznakeFigura(odigraniPotez(1,1) -
                        1,odigraniPotez(1,2) - 1),s3) == 1
180                         s = strcat(s,'+');
181                     end
182                 end
183             else
184                 if strcmp(OznakeFigura(odigraniPotez(1,1) - 1,
                    odigraniPotez(1,2) - 1),s3) == 1 || ...
185                 strcmp(OznakeFigura(odigraniPotez(1,1) -
                    1,odigraniPotez(1,2) + 1),s3) == 1
186                     s = strcat(s,'+');
187                 end
188             end
189         else
190             if odigraniPotez(1,2) == 1
191                 if odigraniPotez(1,1) == 8
192                 else
193                     if strcmp(OznakeFigura(odigraniPotez(1,1) +
                        1,odigraniPotez(1,2) + 1),s3) == 1
194                         s = strcat(s,'+');
195                     end
196                 end
197             elseif odigraniPotez(1,2) == 8
198                 if odigraniPotez(1,1) == 8
199                     else
200                     if strcmp(OznakeFigura(odigraniPotez(1,1) +
                        1,odigraniPotez(1,2) - 1),s3) == 1
201                         s = strcat(s,'+');
202                     end
203                 end
204             else
205                 if strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) + 1),s3) == 1 || ...
206                 strcmp(OznakeFigura(odigraniPotez(1,1) + 1,
                    odigraniPotez(1,2) - 1),s3) == 1
207                     s = strcat(s,'+');
208                 end
209             end
210         end
211     end %KRAJ - provjera - pijun
212 elseif odigranaFigura=='Q'
213     s = strcat('Q',s1);
214     provjeraStupcaIRetka
215     dijagonalnaProvjera
216 elseif odigranaFigura == 'K'
217     s = strcat('K',s1);
218 elseif odigranaFigura == 'N'
219     s = strcat('N',s1);
220     if (odigraniPotez(1,1)-2) >= 1 && (odigraniPotez(1,2)-1) >= 1
        %položaj gore -> lijevo
```

```
221             if strcmp(OznakeFigura(odigraniPotez(1,1)-2,odigraniPotez(1,2)-1),s3)
222                 s = strcat(s,'+');
223             end
224         end
225         if (odigraniPotez(1,1)-2) >= 1 && (odigraniPotez(1,2)+1) <= 8
            %položaj gore -> desno
226             if strcmp(OznakeFigura(odigraniPotez(1,1)-2,odigraniPotez(1,2)+1),s3)
227                 s = strcat(s,'+');
228             end
229         end
230         if (odigraniPotez(1,1)-1) >= 1 && (odigraniPotez(1,2)-2) >= 1
            %položaj lijevo -> gore
231             if strcmp(OznakeFigura(odigraniPotez(1,1)-1,odigraniPotez(1,2)-2),s3)
232                 s = strcat(s,'+');
233             end
234         end
235         if (odigraniPotez(1,1)-1) >= 1 && (odigraniPotez(1,2)+2)<= 8
            %položaj desno -> gore
236             if strcmp(OznakeFigura(odigraniPotez(1,1)-1,odigraniPotez(1,2)+2),s3)
237                 s = strcat(s,'+');
238             end
239         end
240         if (odigraniPotez(1,1)+1) <= 8 && (odigraniPotez(1,2)-2) >= 1
            %položaj lijevo -> dolje
241             if strcmp(OznakeFigura(odigraniPotez(1,1)+1,odigraniPotez(1,2)-2),s3)
242                 s = strcat(s,'+');
243             end
244         end
245         if (odigraniPotez(1,1)+1) <= 8 && (odigraniPotez(1,2)+2) <= 8
            %položaj desno -> dolje
246             if strcmp(OznakeFigura(odigraniPotez(1,1)+1,odigraniPotez(1,2)+2),s3)
247                 s = strcat(s,'+');
248             end
249         end
250         if (odigraniPotez(1,1)+2) <=8 && (odigraniPotez(1,2)-1) >= 1
            %položaj dolje -> lijevo
251             if strcmp(OznakeFigura(odigraniPotez(1,1)+2,odigraniPotez(1,2)-1),s3)
252                 s = strcat(s,'+');
253             end
254         end
255         if (odigraniPotez(1,1)+2) <= 8 && (odigraniPotez(1,2)+1) <= 8
            %položaj dolje -> desno
256             if strcmp(OznakeFigura(odigraniPotez(1,1)+2,odigraniPotez(1,2)+1),s3)
257                 s = strcat(s,'+');
258             end
259         end
260     elseif odigranaFigura == 'B'
261         s = strcat('B',s1);
262         dijagonalnaProvjera
263     elseif odigranaFigura == 'R'
264         s = strcat('R',s1);
265         provjeraStupcaIRetka
```

```
266   end
267   if temp3 == 1 %zamjena ("jedenje") figura
268       if odigranaFigura(1) == 'P'||'B'||'K'||'N'||'R'||'Q'
269       s = strcat(s(1),'x',s(2:size(s,2)));
270       else
271           s = strcat(s2(1),'x',s);
272       end
273   end
```

Prilog E. *provjeraRohade.m*

```
1     prvaFigura = char(OznakeFigura(odigraniPotez2(1,1), odigraniPotez2(1,2)));
2     drugaFigura = char(OznakeFigura(odigraniPotez4(1,1), odigraniPotez4(1,2)));
3     if strcmp(prvaFigura,'-') == 0
4         if strcmp(drugaFigura,'-') == 0
5             if strcmp(prvaFigura,drugaFigura) == 0
6                 if strcmp(prvaFigura(1),'R') == 1 || strcmp(prvaFigura(1),'K') == 1
7                     if dstrcmp(drugaFigura(1),'R') == 1 ||
                      strcmp(drugaFigura(1),'K') == 1
8                         potez = potez + 1;
9                         if mod(potez,2)~= 0 %br. poteza neparan -> bijeli na potezu
10                            bijeli = 1;
11                        else
12                            crni = 1;
13                        end
14                        if smjerKretanjaBijelihFigura == 0 || 1
15                            if odigraniPotez(1,1) == 2 || 3 &&
                              odigraniPotez3(1,1) == 2 || 3
16                                s = 'O-O';
17                                OznakeFigura{odigraniPotez2(1,1),
                                  odigraniPotez2(1,2)} = '-';
18                                OznakeFigura{odigraniPotez4(1,1),
                                  odigraniPotez4(1,2)} = '-';
19                                if bijeli == 1 && smjerKretanjaBijelihFigura == 0
20                                    OznakeFigura{3,1} = 'RW';
21                                    OznakeFigura{2,1} = 'KW';
22                                elseif crni == 1 && smjerKretanjaBijelihFigura == 0
23                                    OznakeFigura{3,8} = 'RB';
24                                    OznakeFigura{2,8} = 'KB';
25                                elseif bijeli == 1 && smjerKretanjaBijelihFigura ==
1
26                                    OznakeFigura{3,8} = 'RW';
27                                    OznakeFigura{2,8} = 'KW';
28                                elseif crni == 1 && smjerKretanjaBijelihFigura == 1
29                                    OznakeFigura{3,1} = 'RB';
30                                    OznakeFigura{2,1} = 'KB';
31                                end
32                            elseif odigraniPotez(1,1) == 5 || 6 &&
                              odigraniPotez(1,1) == 5 || 6
33                                s = 'O-O-O';
```

```
34                                  OznakeFigura{odigraniPotez2(1,1),
                                    odigraniPotez2(1,2)} = '-';
35                                  OznakeFigura{odigraniPotez4(1,1),
                                    odigraniPotez4(1,2)} = '-';
36                                  if bijeli == 1 && smjerKretanjaBijelihFigura == 0
37                                      OznakeFigura{5,1} = 'RW';
38                                      OznakeFigura{6,1} = 'KW';
39                                  elseif crni == 1 && smjerKretanjaBijelihFigura == 0
40                                      OznakeFigura{5,8} = 'RB';
41                                      OznakeFigura{6,8} = 'KB';
42                                  elseif bijeli == 1 && smjerKretanjaBijelihFigura ==
1
43                                      OznakeFigura{5,8} = 'RW';
44                                      OznakeFigura{6,8} = 'KW';
45                                  elseif crni == 1 && smjerKretanjaBijelihFigura == 1
46                                      OznakeFigura{5,1} = 'RB';
47                                      OznakeFigura{6,1} = 'KB';
48                                  end
49                              end
50                          elseif smjerKretanjaBijelihFigura == 2 || 3
51                              if odigraniPotez(1,2) == 3 || 4 &&
                                odigraniPotez3(1,2) == 3 || 4
52                                  s = 'O-O-O';
53                                  OznakeFigura{odigraniPotez2(1,1),
                                    odigraniPotez2(1,2)} = '-';
54                                  OznakeFigura{odigraniPotez4(1,1),
                                    odigraniPotez4(1,2)} = '-';
55                                  if bijeli == 1 && smjerKretanjaBijelihFigura == 2
56                                      OznakeFigura{1,4} = 'RW';
57                                      OznakeFigura{1,3} = 'KW';
58                                  elseif crni == 1 && smjerKretanjaBijelihFigura == 2
59                                      OznakeFigura{8,8} = 'RB';
60                                      OznakeFigura{8,8} = 'KB';
61                                  elseif bijeli == 1 && smjerKretanjaBijelihFigura ==
3
62                                      OznakeFigura{8,4} = 'RW';
63                                      OznakeFigura{8,3} = 'KW';
64                                  elseif crni == 1 && smjerKretanjaBijelihFigura == 3
65                                      OznakeFigura{1,4} = 'RB';
66                                      OznakeFigura{1,3} = 'KB';
67                                  end
68                              elseif odigraniPotez(1,2) == 6 || 7 &&
                                odigraniPotez(1,2) == 6 || 7
69                                  s = 'O-O';
70                                  OznakeFigura{odigraniPotez2(1,1),
                                    odigraniPotez2(1,2)} = '-';
71                                  OznakeFigura{odigraniPotez4(1,1),
                                    odigraniPotez4(1,2)} = '-';
72                                  if bijeli == 1 && smjerKretanjaBijelihFigura == 2
73                                      OznakeFigura{1,6} = 'RW';
74                                      OznakeFigura{1,7} = 'KW';
75                                  elseif crni == 1 && smjerKretanjaBijelihFigura == 2
```

```
                                    OznakeFigura{8,6} = 'RB';
                                    OznakeFigura{8,7} = 'KB';
                                elseif bijeli == 1 && smjerKretanjaBijelihFigura ==
3
                                    OznakeFigura{8,6} = 'RW';
                                    OznakeFigura{8,7} = 'KW';
                                elseif crni == 1 && smjerKretanjaBijelihFigura == 3
                                    OznakeFigura{1,6} = 'RB';
                                    OznakeFigura{1,7} = 'KB';
                                end
                            end
                        end
                    end
                end
            end
        end
    end
```

Prilog F. *dijagonalnaProvjera.m*

```
    temp5 = 1;
    for i = odigraniPotez(1,1)-1:-1:1 %dijagonalna provjera lijevo gore
        if (odigraniPotez(1,2)-temp5) >= 1
            if strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),s3) == 0 &&
             strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),'-') == 0
                break
            elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),s3) == 1
                s = strcat(s,'+');
            end
        end
        temp5 = temp5 + 1;
    end
    temp5 = 1;
    for i = odigraniPotez(1,1)+1:8 %dijagonalna provjera lijevo dolje
        if (odigraniPotez(1,2)-temp5) >= 1
            if strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),s3) == 0 &&
            strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),'-') == 0
                break
            elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)-temp5),s3) == 1
                s = strcat(s,'+');
            end
        end
        temp5 = temp5 + 1;
    end
    temp5 = 1;
    for i = odigraniPotez(1,1)-1:-1:1 %dijagonalna provjera desno gore
        if (odigraniPotez(1,2)+temp5) <= 8
            if strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),s3) == 0 &&
            strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),'-') == 0
                break
            elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),s3) == 1
```

```
29                  s = strcat(s,'+');
30              end
31          end
32          temp5 = temp5 + 1;
33      end
34      temp5 = 1;
35      for i = odigraniPotez(1,1)+1:8 %dijagonalna provjera desno dolje
36          if (odigraniPotez(1,2)+temp5) <= 8
37              if strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),s3) == 0 &&
                strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),'-') == 0
38                  break
39              elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)+temp5),s3) == 1
40                  s = strcat(s,'+');
41              end
42          end
43          temp5 = temp5 + 1;
44      end
```

Prilog G. *provjeraStupcaIRetka.m*

```
1       for i = 8 - odigraniPotez(1,1):8 % provjera po stupcu
2           if strcmp(OznakeFigura(i,odigraniPotez(1,2)),'-') == 0 &&
strcmp(OznakeFigura(i,odigraniPotez(1,2)),s3) == 0
3               break
4           elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)),s3) == 1
5               s = strcat(s,'+');
6           end
7       end
8       for i = 1:8 - odigraniPotez(1,1) % provjera po stupcu
9           if strcmp(OznakeFigura(i,odigraniPotez(1,2)),'-') == 0 &&
strcmp(OznakeFigura(i,odigraniPotez(1,2)),s3) == 0
10              break
11          elseif strcmp(OznakeFigura(i,odigraniPotez(1,2)),s3) == 1
12              s = strcat(s,'+');
13          end
14      end
15      for i = 8 - odigraniPotez(1,2):8 % provjera po retku
16          if strcmp(OznakeFigura(odigraniPotez(1,1),i),'-') == 0 &&
strcmp(OznakeFigura(odigraniPotez(1,1),i),s3) == 0
17              break
18          elseif strcmp(OznakeFigura(odigraniPotez(1,1),i),s3) == 1
19              s = strcat(s,'+');
20          end
21      end
22      for i = 1:8 - odigraniPotez(1,2) % provjera po retku
23          if strcmp(OznakeFigura(odigraniPotez(1,1),i),'-') == 0 &&
strcmp(OznakeFigura(odigraniPotez(1,1),i),s3) == 0
24              break
25          elseif strcmp(OznakeFigura(odigraniPotez(1,1),i),s3) == 1
26              s = strcat(s,'+');
27          end
28      end
```