Izvajanje meritve časa pri preizkusu z devetimi zatiči

Gal Sajko, Jakob Baumgartner

Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška 25, 1000 Ljubljana, Slovenija

**Povzetek.** V okviru projekta pri predmetu Robotski vid, sva se ukvarjala s problematiko merjenja časa pri izvajanju preizkusa z devetimi zatiči z dominanto in nedominantno roko. Meritve sva izvajala s pomočjo kamere in analize posameznih sličic iz videa. Osredotočila sva se na zaznavanje postavljenih zatičov, zatičov, ki še niso bili postavljeni in roke, ki izvaja nalogo. Na podlagi informacije, pridobljene z analizo sličic, sva določila celoten čas opravljanja naloge, obenem pa tudi čase, potrebne za postavitev posameznih zatičov. Na koncu sva dobljene rezultate tudi ovrednotila.

**Ključne besede:** nevrodegenerativne bolezni, preizkus z devetimi zatiči, robotski vid

**A method of automatization for measurement with Nine Peg Hole Test**

**Abstract.** For the assignment for the subject Robotic Vision, we created a method for automatic measurement of execution of the Nine Peg Hole test. For this we capture video from a camera that is located above the board and use different computer vision methods to detect how many pins are inserted in holes on the board and how many left lying. With analysis of images from video we can measure how long does it take for the person tested to insert each pin and how long does it take to complete the entire test. We can also detect if more than one pin is collected at a time.

**Keywords:** neurodegenerative diseases, nine-hole peg test, robot vision

# Uvod

Preizkus z devetimi zatiči je preizkus, ki se ga pogosto uporablja v medicini, za določevanje različnih nevrodegenerativnih bolezni. Pripomočki, potrebni za ta preizkus so posebna ploščica, ki ima na eni strani prostor za odlaganje zatičev, na drugi strani pa devet lukenj, v katere testiranec postavlja zatiče. Zatiči so na začetku preizkusa na odložišču. Naloga testiranca je, da vsak zatič, enega za drugim, postavi v luknjo. Ko je vseh devet zatičev postavljenih, jih mora, ponovno enega za drugim, postaviti nazaj na odložišče. Glede na čas, ki ga testiranec porabi za izvedbo celotne naloge, se ocenjuje morebitno prisotnost bolezni. V sklopu najine naloge, sva čas merila na podlagi analize sličic iz spletne kamere, ki je bila priključena na računalnik. Za uspešno reševanje celotnega problema, sva ga razdelila na sledeče štiri podprobleme: (i) zaznavanje roke na sceni, (ii) sledenje roki, ko je na sceni, (iii) zaznavanje postavljenih zatičev, ter (iv) zaznavanje ne postavljenih zatičev. Za izvedbo meritve, sva napisala program v programskem jeziku Python. Edini podatek, ki je med delovanjem posredovan programu, je posamezna sličica iz kamere.

# Oprema in postavitev

Za izvedbo preizkusa sva uporabila naslednjo opremo:

* Ploščica iz plastike, natisnjena s 3D tiskalnikom. Na eni strani ima okroglo površino za odlaganje zatičev, na drugi pa devet lukenj, postavljenih v mrežo dimenzije 3x3.
* Leseni zatiči, ki sva jih na vrhu prebarvala z zeleno barvo, za lažjo prepoznavo na sliki.
* Namizna svetilka, postavljena nad ploščico. Z uporabo luči sva zmanjšala vpliv zunanje svetlobe.
* Spletna kamera, ki je povezana z računalnikom, na katerem teče program. Kamera je postavljena navpično nad ploščico na tako višino, da je v zajeti sliki čim manj okolice a kljub temu celotna ploščica.

# Metode

Kot sva že omenila, sva celotni problem razdelila na več podproblemov. Za reševanje vsakega podproblema sva uporabila različne metode obdelovanja in analize slik.

## Zaznavanje roke na sceni

Podatek, ali je roka na sceni ali ne, je pomemben zato, ker od njega zavisi, kdaj se začne oziroma ustavi merjenje časa. V trenutku, ko se roka pojavi na sceni, se merjenje časa začne. Merjenje se ustavi, ko so vsi zatiči odloženi nazaj v odložišče in se roka umakne iz scene, seveda ob predpostavki, da je bilo vseh devet zatičev prej že postavljenih. Informacijo o tem, ali je roka prisotna ali ne, sva določila z odštevanjem dveh zaporednih sličic. Na dobljeni sliki sva izvedla tudi operacijo odpiranja, da sva se znebila šuma.

### Odštevanje sličic

Z operacijo odštevanja sličic, dobimo informacijo o tem, kje se sličici razlikujeta med seboj. Glede na lastnosti preizkusa z devetimi zatiči sva predpostavila, da se roka med izvajanjem naloge ves čas premika, ploščica in ostali objekti na sliki, pa so večinoma pri miru. V idealnih razmerah tako dobimo, ob odštevanju dveh zaporednih sličic, na območju, ki se ni spremenilo, ničelne vrednosti pikslov ter na območju, ki ni enako, kot na prejšnji sličici, neničelne vrednosti. V realnih razmerah pa ob odštevanju na območju brez sprememb, dobimo močno pošumljene vrednosti.

### Odpiranje

Operacija odpiranja je kombinacija dveh morfoloških preslikav, in sicer erozije in dilacije, ki ju izvedemo v točno tem vrstnem redu. Ob izvajanju erozije, bo vrednost piksla na sliki enaka 1 samo v primeru, da so vrednosti vseh pikslov pod jedrom prav tako enake 1. V nasprotnem primeru, bo temu pikslu dodeljena vrednost 0 in informacija, ki jo nosi, bo izginila. Dilacija je ravno obratni proces, kot erozija. Vrednost piksla bo ostala 1 v primeru, da je vsaj en piksel pod jedrom enak 1. Z izvajanjem operacij zaporedno, torej najprej izvedemo erozijo in potem na dobljeni sliki še dilacijo, se učinkovito znebimo šuma tipa sol-poper, ki se je pojavil ob odštevanju sličic.

### Končni rezultat

Po izvedbi zgornjih dveh metod, sva dobila sliko, ki je bila črna na območjih, kjer ni bilo sprememb in bela, na območjih sprememb. Ker je spremembe povzročala samo roka, sva lahko iz te slike brez težav določila približno pozicijo središča roke ter njeno prisotnost na sceni.

Opomba: glej Slika1.jpg v mapi Priloge

## Zaznavanje postavljenih zatičev

Štetje postavljenih zatičev na vsaki sličici nam pove, kako testiranec napreduje s svojo nalogo. Naloga je lahko uspešno opravljena samo v primeru, da je v nekem trenutku med izvajanjem, postavljenih vseh devet zatičev ter, da je število postavljenih zatičev na koncu naloge enako 0. Ker je kamera postavljena tako, da snema sceno iz ptičje perspektive, imajo postavljeni zatiči na sličicah obliko kroga. Dimenzije zatičev so nam znane v naprej in se med izvajanjem naloge ne spreminjajo. Te lastnosti sva uporabila za implementacijo metode Houghove transformacije in določanje njenih parametrov, s katero lahko učinkovito prepoznamo krožne strukture na sliki.

### Houghova transformacija

Houghova transformacija je zelo uporabna metoda za iskanje krožnic na binarni sliki. Funkcijo kroga lahko parametriziramo, ter tako dobimo tri parametre: *r –* radij ter *a* in *b –* koordinati središča, ki opisujejo krožnico in tvorijo tri dimenzionalni parametrični prostor. Ko parametrični prostor diskretiziramo, dobimo akumulatorske celice. Vrednost vsake akumulatorske celice na začetku postavimo na 0. V procesu iskanja krožnic spreminjamo vrednosti parametrov središča kroga in za vsako potencialno točko na krožnici izračunamo vrednost radija. Nato vrednost akumulatorske celice pri tem radiju in središču povečamo za 1. Na koncu procesa pogledamo vrednosti akumulatorskih celic ter tako dobimo najbolj očitne krožnice [1]. Iskanje krožnic na sliki sva omejila na območje z luknjami za postavljanje zatičev. Prav tako sva omejila območje radija, da je približno ustrezal dimenzijam zatiča ter tako preprečila iskanje prevelikih ali premajhnih krogov.

### HSV prostor slike

S Houghovo transformacijo sva dobila potencialne lokacije postavljenih zatičev. Vendar pa samo to ni zadostovalo za zanesljivo določanje postavljenih zatičev, saj je ta metoda zaznala krog tako na območju postavljenega zatiča, kot na območju praznih lukenj. Zato sva vsa potencialna središča krogov, dobljenih s Houghovo transformacijo, preverila še v HSV prostoru slike. Za lažje razpoznavanje dejansko postavljenih zatičev, sva njihove vrhove pobarvala z zeleno barvo. Tako sva lahko preverila, ali H komponenta piksla v središču zaznanega kroga ustreza zeleni barvi ali ne. V primeru, da je barva ustrezala, je program lahko potrdil postavljen zatič.

Opomba: glej Slika2.jpg v mapi Priloge

## Zaznavanje ne postavljenih zatičev

### Priprava

Pri štetju posameznih zatičev, ki se nahajajo na odlagalni površini sva iz slike najprej izrezala le območje odlagališča, ter znižala njegovo ločljivost, da metoda poteka hitreje. Nato sva s pomočjo konvolucije z Gaussovim filtrom sliko nekoliko zgladila in na dobljeni sliki izvedla pretvorbo v HSV barvni prostor. S pomočjo binarnega filtriranja H komponente zapisa slike, sva pridobila odtise zatičev, ki pa so zaradi nižje resolucije in glajenja le okvirni, ter se med seboj prelivajo.

### Detekcija

Nato sva za vsako točko na sliki preverila, če je njena vrednost različna od nič, to pomeni, da se nahaja na enemu od odtisov zatičev. Če je bila preverba pozitivna, postane to center možne zaznave zatiča. Tedaj za korake po pet stopinj detekcijski pravokotnik rotirava in preverjava preostale točke, ki bi se nahajale na potencialnem zatiču, ki se nahaja na plošči, s centrom v dani točki in ležeč pod kotom glede na vodoravnico. To izvajava postopoma, da v primeru negativne zaznave zmanjšava število operacij. Rezultat je več sto potencialnih pozicij zatičev.

### Izključevanje

Zaradi nenatančnih obrisov, ki so posledica senc in šuma, detekcijski pravokotnik ne mora biti popolnoma ujemajoč se z velikostjo zatičev na površini. Posledično dobimo več sto zaznav, ki jih sprva filtriramo tako, da odstranimo zaznave, ki imajo center v bližnjih točkah in ležijo pod podobnim kotom. Nato preverimo, če se zaznani zatiči med seboj prekrivajo. To izvedemo z detekcijo sekanja premic stranic med pari zaznanih zatičev. V primeru, da se prekrivajo izločimo tisto zaznavo, ki se seka z več preostalimi zatiči. To izločanje izvajamo dokler se zaznani zatiči med seboj ne prekrivajo več in imamo zaznane le lokacije dejanskih zatičev.

Opomba: glej Slika3.png v mapi Priloge

# Implementacija programa

Zgoraj opisane metode sva implementirala s programskim jezikom Python in knjižnicama Numpy in OpenCV, ter jih združila v program, ki omogoča izvedbo celotnega preizkusa.

## Zasnova programa

Program je zasnovan tako, da omogoča meritve časa opravljanja preizkusa za obe roki. Na koncu nam poda čas za vsako roko ter razliko med tema dvema časoma. Poleg tega sproti spremlja čase, porabljene za postavitev posameznih zatičev, ter na koncu poda informacije o najhitrejšem, najpočasnejšem ter povprečnem času za postavljanje posameznega zatiča.

### Potek programa

Najprej izvajamo meritev z levo roko. Zato ploščico postavimo tako, da je odložišče na desni in luknje za postavljanje zatičev na levi strani. Prepričamo se, da so vsi zatiči v odložišču. Z uporabo računalniške miške, na sliki izberemo zgornjo levo ter spodnjo desno luknjo, ter na ta način določimo področje na sliki, kjer se nahajajo luknje za postavljanje zatičev. Nato na tej sliki označimo še zgornji levi ter spodnji desni kot ploščice. S pritiskom na tipko ENTER povemo programu, da smo pripravljeni na začetek merjenja. Z levo roko začnemo postavljati zatiče v luknje. Ko smo uspešno postavili vseh devet zatičev, nam program javi, da so vsi zatiči postavljeni. Nato začnemo zatiče nositi nazaj v odložišče. Ko smo končali, umaknemo roko iz scene in meritev za levo roko se zaključi. Izpiše se čas celotnega poteka ter statistika časov za postavljanje posameznih zatičev. Nato meritev ponovimo z desno roko, s tem da prej ploščico obrnemo ter na novo označimo območja lukenj ter ploščice. Ko končamo meritev tudi z desno roko, se nam poleg že omenjenih časov izpiše tudi razlika v času med levo in desno roko.

## Podrobnosti implementacije posameznih metod

Za pridobivanje želenih informacij iz sličic, ki jih kamera posreduje programu, je potrebno vsako sličico obdelati. Vendar pa vsak podproblem zahteva specifično obdelavo oziroma pripravo slike.

### Priprava slike za določanje postavljenih zatičev

Določanje postavljenih zatičev sva izvedla v dveh korakih. Najprej sva na sivinski sliki poiskala vse potencialne kroge, ki bi lahko predstavljali postavljene zatiče. V ta namen sva originalno sličico najprej pretvorila v sivinsko. Nato sva nad sivinsko sličico izvedla gama preslikavo in na ta način izboljšala kontrast. Na koncu sva sliko še zgladila z Gaussovim glajenjem ter tako zmanjšala učinke šuma. Na dobljeni sliki, sva na območju z luknjami, ki ga ročno označi uporabnik, iskala vse potencialne kroge. Hkrati sva originalno sličico preslikala tudi v HSV prostor, v katerem sva preverila ustreznost barve v središču najdenih krogov.

### Minimiziranje napake zaradi nezanesljivosti iskanja krogov

Zaradi prisotnosti različnih motilnih dejavnikov na originalni sličici se je lahko zgodilo, da je na eni sličici postavljen zatič bil zaznan, na naslednji pa ne. Da sva se izognila temu nihanju, sva spremljala število potencialno postavljenih zatičev skozi tri zaporedne sličice. V primeru, da je bil postavljen zatič zaznan v dveh od treh zaporednih sličicah, sva ga štela za potrjenega in spremenila vrednosti spremenljivke, ki šteje postavljene zatiče.

### Izbira trenutkov za štetje postavljenih zatičev in merjenje časa za postavitev posameznega zatiča

Zaradi narave preizkusa, kjer se roka premika od leve proti desni in pobira ter postavlja zatiče se dogaja, da so približno polovičen del časa postavljeni zatiči zakriti z roko. Zato sva opisano zaznavanje postavljenih zatičev izvajala samo v trenutkih, ko roka ni bila nad postavljenimi zatiči. Na podoben način, sva merila tudi čas postavitve posameznega zatiča. Ta čas se meri od trenutka, ko je bilo zaznano povečanje števila postavljenih zatičev, do trenutka, ko je to število ponovno povečano, z izjemo časa prvega zatiča, ki se meri od trenutka, ko je uporabnik začel z meritvijo. V izogib temu, da bi uporabnik postavil dva zatiča naenkrat, se je morala roka po povečanju vmes premakniti nazaj nad odložišče.

### Priprava slike in način zaznavanje roke

Za zaznavanje roke sva najprej originalno sličico obrezala tako, da nama je ostala samo sličica ploščice, brez podlage. To sličico sva nato pretvorila v sivinsko ter jo odštela od prejšnje, na zgoraj opisan način. Za odločanje, ali je roka na sceni ali ne, sva štela količino svetlih pikslov, ki predstavljajo razlike med dvema zaporednima sličicama. V primeru, da je bilo to število nad določenim pragom, je bila roka zaznana. Približen položaj roke sva določila tako, da sva poiskala lokacijo piksla na sredini tega belega področja, ki je na sliki razlik predstavljalo roko.

### Zaznavanje ne postavljenih zatičev

Zaznavanje zatičev ki so na odlagalni površini ne poteka v realnem času, saj algoritem ni optimiziran za dovolj hitro delovanje. Izvede se naknadno, po končanemu merjenju z zgoraj opisanimi metodami, ter le na posameznih slikah, ki se posnamejo med izvajanjem meritev. Slike se zajamejo vsakič po tem, ko roka zapusti območje odložišča in nese zatič v eno izmed prostih lukenj. Z zaznavanjem ne postavljenih zatičev lahko dobimo dodatne informacije, kot so recimo podatek ali je ob določenem času, uporabnik imel v roki več kot le en zatič. Hitrost zaznave je močno odvisna od števila preostalih zatičev in njihovi medsebojni oddaljenosti na plošči. Pri vseh zatičih na plošči lahko pride do napačne zaznave, saj je pri izločanju zatičev pomembno v kakšnem vrstnem redu se tega koraka lotimo. Temu se najlažje izognemo z uvedbo večih ponovitev in premešanju potencialno zaznanih zatičev. Algoritem ni implementiran za prekrivajoče se zatiče.

# Validacija rezultatov in analiza problemov

Program ločeno izvaja dva sklopa meritev. Prvi je čas celotnega procesa postavljanja in podiranja zatičev, drugi pa časi postavljanja posameznih zatičev. Zato sva na podlagi večih ponovitev meritev ločeno validirala rezultate obeh sklopov. Zaradi nekaterih ne odpravljenih hroščev v programu, se včasih lahko zgodi, da program ne prepozna vseh postavljenih zatičev in je posledično meritev neuporabna. Posebej velja izpostaviti hrošč v programu, ki neposredno vpliva na izvajanje meritve. Program je zasnovan tako, da v trenutku, ko zazna 9 postavljenih zatičev, postavi zastavico in to javi uporabniku. Če ta zastavica ni postavljena, se čas merjenja ne ustavi tudi, ko je število postavljenih zatičev ponovno enako 0. Do problema pride, če uporabnik postavi vseh devet zatičev, program pa jih zaradi takšnih ali drugačnih razlogov ne zazna, ali pa za zaznavo potrebuje nekaj časa. Tako se čas izvedbe podaljša. To sicer ne vpliva neposredno na natančnost meritve izvedbe, vpliva pa na brezhibno delovanje programa, saj mora uporabnik po postavitvi devetega zatiča počakati, da program potrdi, da je uspešno zaznal 9 postavljenih zatičev. Drugi večji problem se pojavi pri merjenju časov za postavljanje posameznega zatiča. Čas se meri med dvema povečanjema vrednosti spremenljivke, ki shranjuje število postavljenih zatičev. To pa je ponovno odvisno od zanesljivosti in hitrosti razpoznavanja postavljenih zatičev. Lahko se zgodi, da program novega postavljenega zatiča ne prepozna. Zaradi spremenljive svetlobe se lahko zgodi tudi, da program prepozna postavljen zatič na mestu, kjer je v resnici prazna luknja v ploščici. Za uspešno opravljeno meritev časov za postavljanje posameznih zatičev bi potrebovali 9 popolnih prepoznav postavljenih zatičev za vsako roko. Žal najin program zaenkrat take zanesljivosti še ne zagotavlja.

## Validacija merjenja celotnega časa postavljanja zatičev

Pri validaciji merjenja celotnega časa sva izvedla deset meritev in si beležila izmerjene čase. Celoten proces sva tudi posnela in iz videa določila še referenčne oziroma pravilne čase. Izmerjene in referenčne čase sva nato primerjala in določila povprečno napako. Poleg tega sva na podlagi videa določila tudi povprečni čas, ki ga je uporabnik moral počakati, da se je omenjena zastavica, ki potrdi, da je postavljenih vseh 9 zatičev, postavila. V spodnji tabeli so prikazani rezultati validacije:

Tabela 1: Izmerjeni čas, referenčni čas, napaka, napaka zastavice in povprečni napaki, vse v sekundah.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Meritev** | **Izmerjeni čas** | **Referenčni čas** | **Napaka** | **Napaka zastavice** |
| 1 | 33,21 | 32,33 | 0,88 | 0,43 |
| 2 | 33,34 | 31,65 | 1,69 | 0,18 |
| 3 | 31,4 | 30,58 | 0,82 | 1,00 |
| 4 | 29,36 | 28,70 | 0,66 | 0,10 |
| 5 | 31,55 | 31,25 | 0,30 | 1,04 |
| 6 | 29,31 | 29,04 | 0,27 | 0,10 |
| 7 | 30,45 | 30,21 | 0,24 | 0,93 |
| 8 | 32,06 | 31,78 | 0,28 | 1,42 |
| 9 | 28,91 | 28,52 | 0,39 | 0,24 |
| 10 | 27,93 | 27,30 | 0,63 | 0,10 |
| **Povprečje** | 30,75 | 30,14 | 0,61 | 0,55 |

## Validacija merjenja časa za postavljanje posameznega zatiča

Za validacijo tega časa sva iz posnetega videa izbrala primer opravljanja poskusa z eno roko, kjer je program uspešno prepoznal vse postavljene zatiče ter uspešno izmeril vse posamezne čase. Zaradi zgoraj opisanih problemov je ta validacija nekoliko pristranska, zato je potrebno opomniti, da je tukaj prikazan samo tisti primer, kjer program deluje brezhibno. Po drugi strani pa se je potrebno zavedati, da v primeru napake v programu celotna meritev časov za postavljanje zatičev postane brezpredmetna, saj en čas zavisi od drugega in se napaka prenaša do konca meritve. Zato validacija takega primera, ko se v programu pojavi napaka, ne bi imela smisla. Program je uspešno izvedel vse meritve posameznih časov v 50% meritev. Natančneje, uspešno je izvedel vse meritve leve roke, pri vseh meritvah desne roke, pa je prišlo do napake. Sklepava, da vzrok za to tiči v neenakomerni osvetlitvi in sencah na sliki, kar otežuje zaznavanje položaja roke ter postavljenih zatičev.

Tabela 1: Izmerjeni posamezni čas, referenčni posamezni čas, napaka in povprečna napaka, vse v sekundah.

|  |  |  |  |
| --- | --- | --- | --- |
| **Zatič** | **Izmerjeni čas** | **Referenčni čas** | **Napaka** |
| 1 | 2,85 | 2,05 | 0,8 |
| 2 | 2,04 | 1,88 | 0,16 |
| 3 | 2,35 | 1,39 | 0,96 |
| 4 | 2,12 | 1,85 | 0,27 |
| 5 | 1,33 | 1,78 | 0,45 |
| 6 | 2,01 | 1,52 | 0,49 |
| 7 | 3,44 | 2,79 | 0,65 |
| 8 | 2,33 | 2,36 | 0,03 |
| 9 | 1,72 | 1,70 | 0,02 |
| **Povprečje** | 2,24 | 1.92 | 0,43 |

## Komentar validacij

Pri validaciji merjenja celotnega časa vidimo, da znaša povprečna napaka le 0.61 sekunde. Glede na povprečni referenčni čas, ki znaša 30.14 sekunde je napaka 2%, kar bi ocenila za precej dober rezultat. Sama metoda merjenja pa zaradi postavljanja zastavice podaljša čas meritve za povprečno 0.55 sekunde. Sklepava, da bi posamezna meritev trajala 0.55 sekunde manj, če program ne bi vseboval omenjenega hrošča, vendar pa to ne vpliva na natančnost same meritve.

Pri validaciji merjenja časov za postavljanje posameznega zatiča opazimo precej večjo procentualno napako, ki tukaj znaša povprečno 22%. To napako lahko pripiševa tudi temu, da se čas ne izmeri v trenutku, ko je zatič postavljen ampak šele, ko se roka umakne in program zatič prepozna. To potrjuje tudi dejstvo, da so referenčni časi praviloma krajši, kot izmerjeni časi.

Opomba: glej Video1.mp4 za primer uspešne meritve za eno roko in Video2.mp4 za primer neuspešne meritve. Rdeča pika predstavlja približno lokacijo roke.

# Zaključek

Najin cilj je bil izdelati aplikacijo, ki bi čim bolje merila čas postavljanja in podiranja zatičev ter čase postavljanja posameznih zatičev. Pri tem sva implementirala tudi grobo sledenje poziciji roke. Glede na rezultate validacije lahko prvi del naloge označiva za uspešno opravljen, medtem ko pri drugem delu še ostaja precej prostora za izboljšavo. Ključna pomanjkljivost, ki povzroča težave pri merjenju časov je nezanesljivo razpoznavanje postavljenih zatičev. Prav tako bi bila dobrodošla izboljšava pri osvetljevanju scene. Potrebno bi bilo postaviti (več) luči tako, da bi izničila vse sence, ki se pojavljajo zaradi roke in postavljenih zatičev. Idealno bi bilo zagotoviti tako okolje, kjer je osvetljenost ves čas konstantna in ni odvisna od zunanjih dejavnikov, kot so vreme, ura v dnevu, itd.

Kljub temu sva med izdelovanjem aplikacije pridobila veliko novih, predvsem praktičnih, znanj iz področja obdelave in analize slik.

# Literatura

1. Likar Boštjan, “Biomedicinska slikovna informatika in diagnostika”, Založba FE in FRI, str. 161-164