Vizijski sustav za prepoznavanje objekata prema boji

Antić, Ivan

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

IVAN ANTIĆ

Zagreb, 2016. godina.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

VIZIJSKI SUSTAV ZA PREPOZNAVANJE OBJEKATA PREMA BOJI

Mentori:

Prof. dr. sc. Mladen Crneković, dipl. ing.

Student:

Ivan Antić

Zagreb, 2016. godina.

Izjavljujem da sam ovaj rad izradio samostalno koristeći stečena znanja tijekom studija i navedenu literaturu.

Zahvaljujem se dragom Bogu na prilici koju mi je dao da mogu studirati i što su mi roditelji omogućili studij. Zahvaljujem se svojoj supruzi Miji, sestrama i braći, te prijateljima. Neizmjerno hvala na stručnom i strpljivom vodstvu mentora gosp. Mladena Crnekovića bez čijih smjernica i uputa ne bih uspio obraditi ovu tematiku.

Zahvaljujem se na poseban način kolegama koji su interesom i sugestijama doprinijeli također realizaciji ovog rada: Boži Poljaku, Krešimiru Duvnjaku i Domagoju Antiću.

Ivan Antić

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za diplomske ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum 12 -05- 2016	Prilog
Klasa: 602-04/16-6/3	
Ur.broj: 15-1703-16-201	

DIPLOMSKI ZADATAK

Student: **IVAN ANTIĆ** Mat. br.: 0035177574

Naslov rada na hrvatskom jeziku: **VIZIJSKI SUSTAV ZA PRAĆENJE OBJEKATA PREMA BOJI**

Naslov rada na engleskom jeziku: **VISION SYSTEM FOR OBJECT COLOR TRACKING**

Opis zadatka:

Osnovni preduvjet za inteligentno ponašanje mobilnih robota je spoznaja o prostoru u kojem se kreću, klasifikacija situacije u kojoj se nalaze te posjedovanje strategija izvršavanja zadataka. Programiranje takvog sustava moguće je samo u programskim jezicima koji su orijentirani zadatku, a najbolje informacije takvim programskim jezicima daju vizijski sustavi.

Za potrebe vođenja mobilnih robota potrebno je vizijskim sustavom pratiti objekte prema boji, a podatke o prepoznatome serijskom komunikacijom slati na drugo računalo ili drugom programu.

U radu je potrebno:

- ostvariti funkciju istovremenog prepoznavanja i praćenja crvene, zelene i plave boje,
- informaciju o prepoznatom (boja, položaj i veličina) slati drugoj aplikaciji,
- druga aplikacija treba grafički prikazati rezultate prepoznavanja u realnom vremenu.

Zadatak zadan:	Rok predaje rada:	Predviđeni datum obrane:
10. ožujka 2016.	12. svibnja 2016.	18., 19. i 20. svibnja 2016.

Zadatak zadao: Predsjednik Povjerenstva:

Prof.dr.sc. Mladen Crneković Prof. dr. sc. Franjo Cajner

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

SAŽETAK

Cilj ovog rada je prikazati kako pomoću programskog paketa „*Delphi*“-a izvršiti obradu slike s web-kamere u svrhu prepoznavanja objekata prema boji. Korisnik mišem na slici, dobivenoj pomoću web-kamere, izvrši odabir objekta određene boje (crvene, plave ili zelene) za kojeg želi saznati njegove koordinate na slici te veličinu. U prepoznavanju objekta na slici je korištena transformacija boje piksela iz *RGB*-a u *HSV* sustav. Sintakse korištene u obradi slike su opisane u radu. Naposljetku, navedene su i upute kako koristiti sučelje programa napisanog u *Delphi*-u za prepoznavanje objekta prema boji.

Ključne riječi: RGB, HSV, obrada slike, prepoznavanje objekta na slici po boji, *Delphi*

SUMMARY

The purpose of this project was to demonstrate how to process images captured by web-camera using *Delphi* software package in order to recognize object in image by color. By clicking a mouse on image, user selects an object of certain color (red, green or blue) in an image captured by web-camera in order to obtain object's coordinates and size in the image. The transformation of pixel color from RGB to HSV system is used in recognition of object in an image. Syntaxes used in picture processing are described. Lastly, object recognition by color usage instructions for the programe interface created by *Delphi* software are listed.

Key words: RGB, HSV, image processing, recognition of object by color, Delphi

1. UVOD

Osobitost ljudskog vida je da u slici relativno jednostavno prepoznaje određene, ranije naučene strukture (npr. predmete, ljude, životinje), što nije slučaj s računalnim vidom koji sliku registrira kao matricu popunjenu RGB vrijednostima (*eng. Red Green Blue – Crveno Zeleno Plavo*). Glavni cilj računalnog vida jest implementirati računalima vid što sličniji ljudskom. Slika 1. prikazuje jedno od mogućih ljudskih interpretacija slike na kojoj su prikazane različite strukture: pozadina, šešir, žena, zrcalo i odraz u zrcalu, usporedno sa Slikom 2. koja prikazuje moguću računalnu interpretaciju iz koje je čak i čovjeku nemoguće zaključiti o kakvoj se slici radi. [2]

Slika 1. Crno-bijela slika

Slika 2. Računalna interpretacija Slike 1.

Glavni problem računalnog vida jest na koji način obraditi podatke o slici kako bi iz njih dobili korisne informacije. S obzirom da su u računalnom vidu, podaci o slici zapisani kroz boju u

pikselu, slijedi da je jedan od mogućih pristupa traženje podataka o koncentraciji određenog pigmenta boje u polju piksela. Takav pristup daje informacije o polju piksela iz kojega se iščitava vrijednost pigmenta boje i zasićenost pigmenta određenom bojom, za odabrani objekt na slici koji računalni vid treba prepoznati. Pristup koji nam je ovdje potreban je traženje grupe polja unaprijed definirane veličine (polje čini grupa piksela), koji sadrže jednaku ili približno jednaku vrijednost pigmenta boje u usporedbi s referentnim uzorkom. Spomenuta polja s obzirom na veličinu narušavaju kvalitetu slike, ali pritom se smanjuje vrijeme obrade slike i povećava brzina osvježavanja starih rezultata koji su se dobili iz prethodne obrade slike.

Ovaj oblik prepoznavanja objekta određene boje na slici može se primijeniti u traženju ili praćenju objekta prilikom navigacije mobilnih strojeva kao što su mobilni roboti i slična autonomna vozila s ugrađenim vizijskim sustavom.

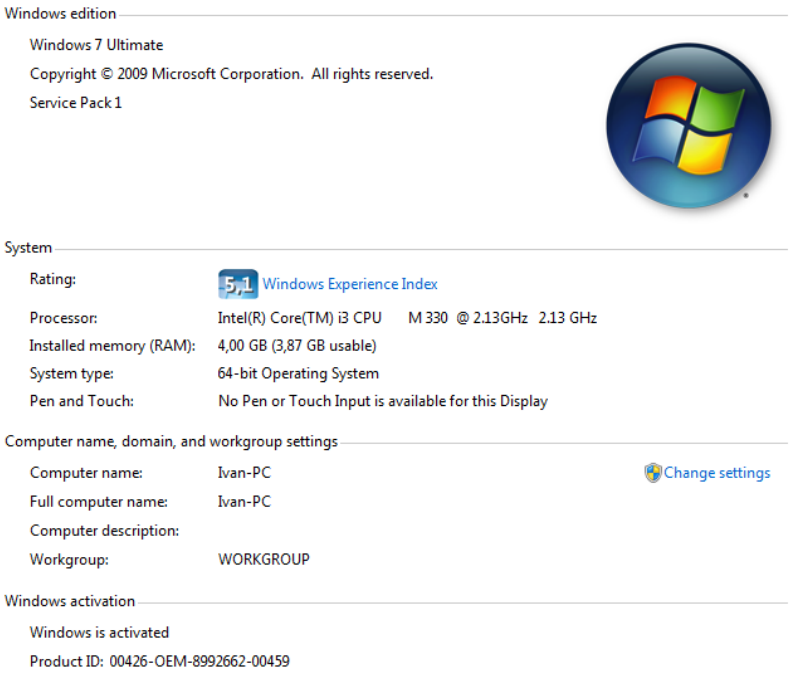
2. RAČUNALNA OPREMA I PROGRAM ZA OBRADU SLIKE

2.1. Web-kamera i operativni sustav

U radu se koristi integrirana web-kamera osobnog prijenosnog računala TOSHIBA - Satellite L650, modela *Calpella* – rezolucije 640x480.

Slika 3. Web-kamera Toshiba prijenosnog računala

Na istoimenom prijenosnom računalu, instaliran je operativni sustav Windows 7. Na Slici 4. su prikazane i ostale pojedinosti operativnog sustava.

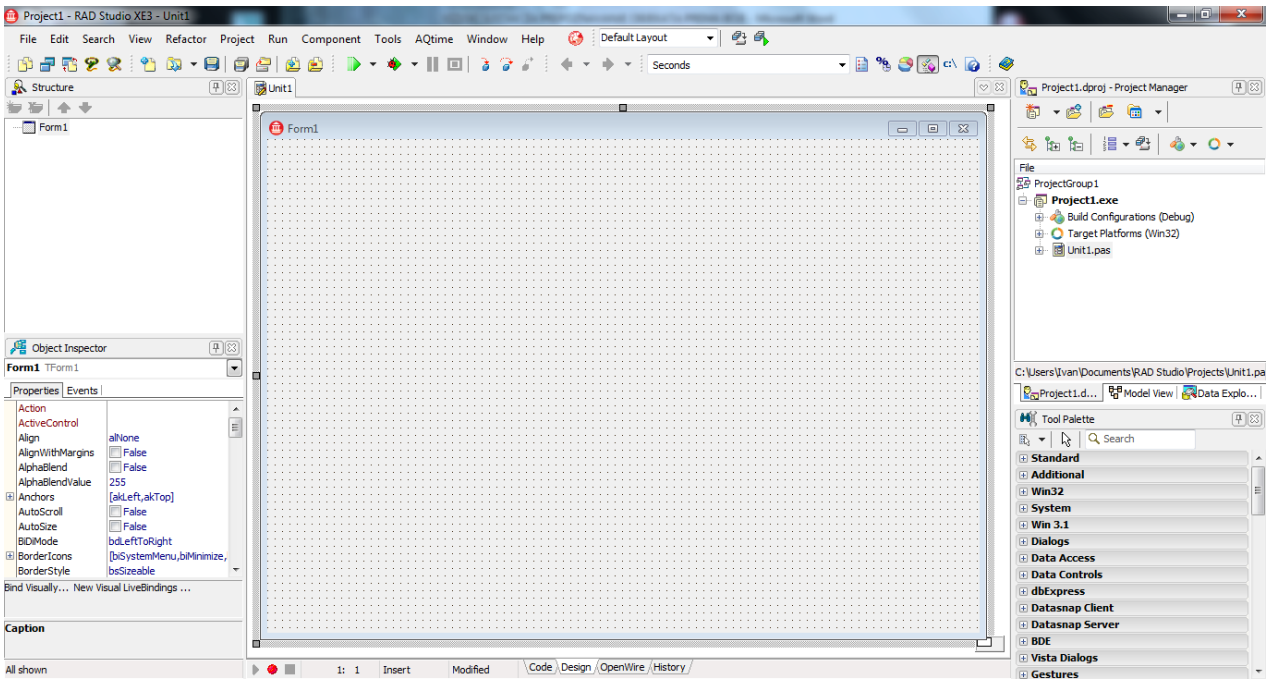


Slika 4. Pojedinosti operativnog sustava Widnows-a 7

2.2. Programsko sučelje „*Delphi*“

Programski paket *Delphi* izdan je do sada u mnogo verzija i izdanja. *Delphi* koji se koristi u ovom radu je u izdanju *Embarcadero RAD Studio* za *Windows*-e u verziji *Delphi XE3*. Programsko okruženje *Delphi*-a je pisano u *Pascal*-u. *Delphi* je oblikovan i uređen za objektno programiranje. Njegov glavni prozor (Slika 5.) podijeljen je u više dijelova, a sačinjavaju ga:

- izborna traka
- u centru prozora je smještena *Form*-a, sa pripadajućim *Unit*-om
- *Structure*: prikazuje kako je ustrojen *Unit* (sadržaj njegovih varijabli, pointera, te drugih *uses*-a (biblioteka) koje se koriste u samome *Unit*-u)
- *Object inspector*: inspektor objekta nudi korisniku da pristupa pojedinim značajkama objekta, koji se uključuje u *Form*-u tj. postavlja na *Form*-u; inspektor objekta sadrži dvije kartice – „Svojstva“ (eng. *Properties*) i „Događaji“ (eng. *Events*) s kojima je povezan objekt
- *Project Manager*: prozor u kojemu je zorno, preko strukture stabla, prikazano s kojim je sve datotekama *Unit* povezan
- *Tool Palette:* paleta alata je izbornik sa svim objektima koji su nam na raspolaganju, kao što su objekti: *Image, Button, Checkbox, Groupbox, Radiobutton, Memo, Label, Timer, VLDSCapture ...*
- *Messages*: prozor koji se pojavljuje ukoliko ima povratnih poruka prema korisniku kada ga *Compiler* obaviještava o nekakvim nepravilnostima u programu ili mu ukazuje na pogrešku u sintaksi koda koju je korisnik napisao



Slika 5. Raspored prozora u *Delphi*-u

3. KOMPONENTE SLIKE

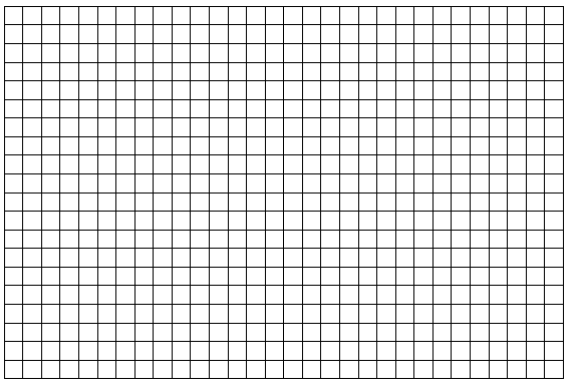
U ovom poglavlju opisane su komponente slike, kako računalo pohranjuje sliku u memoriju, RGB sustav boja, HSV sustav boja, polje piksela.

3.1. Slika u računalu

Sliku kao cjelinu, u kojoj ljudsko oko uočava oblike, boje, linije, grupira objekte sa slike, razlikuje mrtvu prirodu od žive, prepoznaje ljude i životinje te prema prethodno naučenom znanju cjelokupno promatra koju poruku slika prenosi, nije moguće tako jednostavno prenijeti i računalu. U nastavku će biti opisani segmenti slike i način kako računalo prikuplja informacije iz slike.

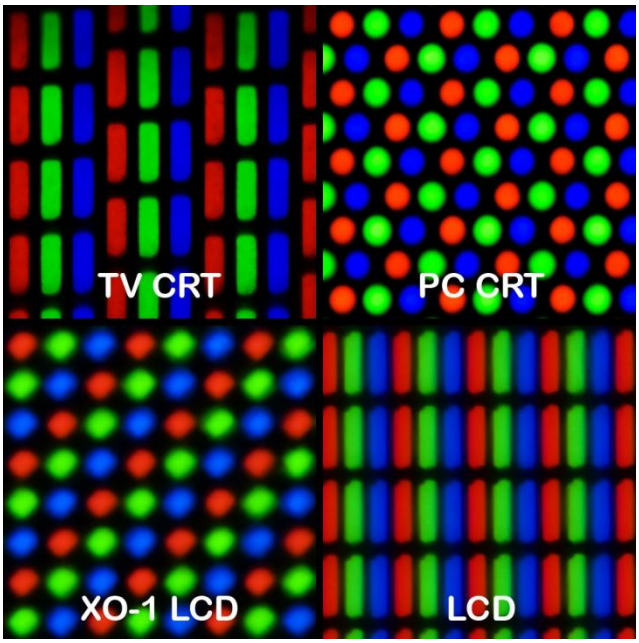
3.1.1. Piksel

Slika kao elektronički zapis u računalu je sačinjena od polja piksela, koja zajedno čine cjelokupnu sliku. Pikseli slike definiraju rezoluciju slike. Slika je skup piksela koji su raspoređeni u stupce i retke. Rezolucija se definira brojem piksela postavljenih u dužinu i visinu slike. Što je veći broj piksela, tj. rezolucija slike, to je veća mogućnost bilježenja više informacija o slici na mjestima promjene: točke, linije, oblika, nijanse boje, i sl.



Slika 6. Primjer rezolucija slike 30x20

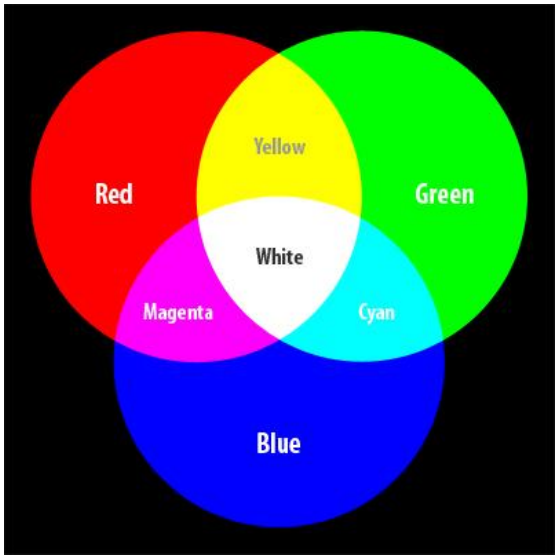
Slika 6. prikazuje izgled i raspored piksela. Izgled piksela nije uvijek pravokutan ili kvadratičan kako se uobičajeno misli. Slika 7. pokazuje neke od mogućih oblika piksela prikazanih pod jako velikim uvećanjem. Rezolucija slike je definirana određenim normama. Rezolucija kamere korištene u ovom radu je *640x480*.



Slika 7. Razni oblici piksela kod različitih uređaja

3.1.2. RGB sustav boja

RGB (kratica od eng. Red, Green, Blue) je zapis boje svakoga piksela. Svaka od ove tri boje ima vrijednost od 0 do 255, tj. zapisuje se sa 8 bitova. Tako ove tri boje daju mogućnost od $\mathbf{2^8 \cdot 2^8 \cdot 2^8 = 2^{24} = 16777216}$ boja i nijansi. Na taj način u memoriji računala svaki piksel slike u sebi nosi zapisanu vrijednost boje koju sačinjavaju crvena, plava i zelena sa svojim udjelom za taj jedan piksel slike.



Slika 8. RGB sustav boja

Konvencionalno je dogovoreno da računalo zapisuje sliku kao medij upravo preko RGB sustava. Ovaj je dogovor usvojen kao pravilo u arhitekturi računala. Slijedi primjer sintakse, u programa *Delphi*-u, pomoću koje će se zapisati vrijednosti RGB za svaki piksel slike u matricu koju ćemo dalje koristiti u obradi slike. Linije programa su:

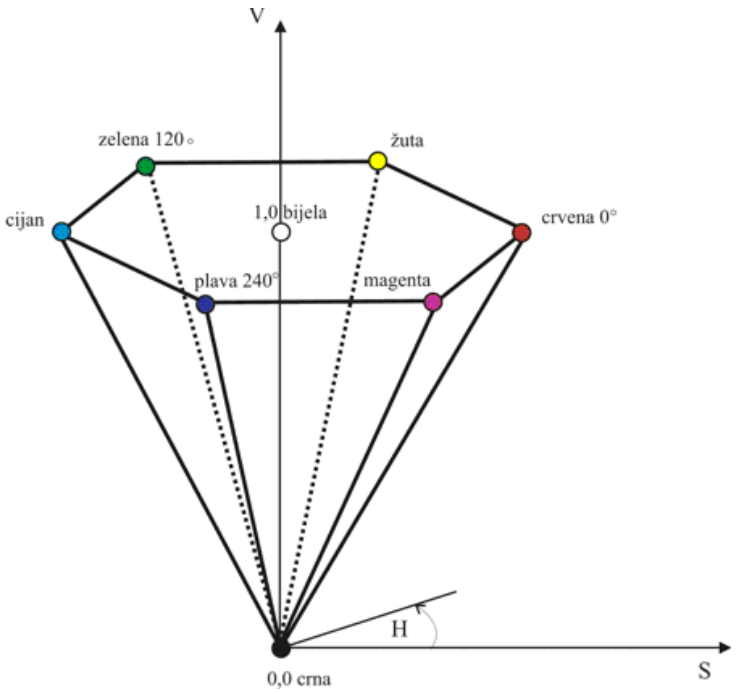
```
procedure slika_u_matricuRGB(slika_kamere: TBitmap);
var
  x,y: integer;
  linija: PRGB;        //Pointer  PRGB služi za učitavanje cijeloga niza vrijednosti crvene, zelene i plave za jedan redak piksela slike
begin
   for y := 0 to 479 do                    //Rezolucija slike po visini je 480 piksela, matrični zapis adrese polja počinje od 0
     begin
       linija := slika_kamere.ScanLine[y];
       for x := 0 to 639 do                //Rezolucija slike po dužini je 640 piksela, matrični zapis adrese polja počinje od 0
         begin
           matricaRGB[y,x].Red := linija[x].rgbRed;
           matricaRGB[y,x].Green := linija[x].rgbGreen;
           matricaRGB[y,x].Blue := linija[x].rgbBlue;
         end;
     end;
   Definiranje_matriceHSV;
end;
```

Definirana je procedura *slika_u_matricuRGB(slika_kamere: TBitmap)* koja pri svome pozivu prima objekt u formatu *.bmp* koju čini slika s kamere. Potom su definirane dvije lokalne varijable *x*, *y* i jedan pointer *linija*. Pomoću dvije ugniježdene *for* petlje u pointer *linija* se preko naredbe *ScanLine* dohvaćaju vrijednosti slike koje su pospremljene u memoriji računala i preko pointera se spremaju u matricu imena *matricaRGB*. Matrica imena *matricaRGB* sada sadrži vrijednosti slike za svaki piksel u kombinaciji crvene, zelene i plave boje. Kako za daljnju analizu i obradu slike ovaj sustav RGB vrijednosti boja piksela nije jednoznačan izvršiti će se transformacija iz RGB sustava boja u HSV sustav boja za svaki piksel čije vrijednosti su upisane u matricu *matricaRGB*.

3.1.3. HSV sustav boja

U potrazi za filterom i kriterijima po kojima bi filter trebao prepoznavati samo jednu boju koju korisnik želi, sustav RGB vrijednosti boja se pokazuje nepraktičan. Naime, jako je teško odrediti uvjete za svaku od tri boje, crvenu, zelenu i plavu, koji bi zadovoljavali kriterij po kojem bi se vršila pretraga određene boje. Stoga će se koristiti HSV sustav određivanja boje piksela. Za to će biti potrebna transformacija vrijednosti boje iz RGB u HSV.

HSV (kratica od eng. *hue*, *saturation*, *value*) model boje bliži je intuitivnom shvaćanju boje. Ovaj model definira se u cilindričnom koordinatnom sustavu. Boje su prikazane u podskupu prostora omeđenom šesterostranom piramidom. Vrh piramide je u ishodištu i odgovara crnoj boji. Vertikalna os V određuje sjajnost boje (*value*). Baza piramide je na razini V=1 što odgovara skupu sjajnih boja. Nijansa (*hue*) određena je kutem zakreta H oko vertikalne osi V. Kutovi zakreta komplementarnih boja razlikuju se za 180°. Radijalna udaljenost od osi V određuje razinu zasićenosti boje, S (*saturation*). HSV model boje prikazan je na Slici 9.[3]



Slika 9. Grafički prikaz HSV sustava boje

Kako je vidljivo iz slike 9. sve nijanse (hue-ovi) boje su jednoznačno određeni, te osim nijanse H imamo još dva parametra koja nam određuju boju, zasićenost S (saturation) i sjajnost V (value) boje.

Sada je potrebno, za postojeću matricu s RGB vrijednostima slike koji su pohranjeni u matricu matricaRGB, za svako polje matrice izvršiti transformaciju iz RGB u HSV sustav vrijednosti boje te tako dobiti matricu matricaHSV. Naredba za transformaciju iz RGB u HSV je napisana pomoću dvije *for* petlje u kojoj se za svaki piksel matrice matricaRGB poziva funkcija preslikavanja imena function RGB2HSV (matricaRGB : TRGB) : THSV;.

```
procedure Definiranje_matriceHSV;
var
  xx, yy: integer;
begin
  for yy := 0 to 479 do
    for xx := 0 to 639 do
      begin
matricaHSV[yy,xx] := RGB2HSV(matricaRGB[yy,xx]);
      end;
Robusna_matricaHSV7x7;
end;
```

Na kraju procedure slika_u_matricuRGB(slika_kamere: TBitmap); pozvana je procedura u kojoj se vrši upis HSV vrijednosti u novu matricu za svaki piksel. U proceduri Definiranje_matriceHSV za određivanje HSV vrijednosti piksela za svaki piksel slike poziva se funkciju imena RGB2HSV (matricaRGB : TRGB) : THSV; po kojoj će se izvršiti transformacija vrijednosti piksela iz RGB u HSV sustav boja. Na početku je potrebno odrediti maksimalnu i minimalnu vrijednost od triju boja u RGB sustavu. Potom se iz tih dviju vrijednosti dobije delta.

$$\boldsymbol{f(R,G,B) = max\,(R,G,B)}$$
$$\boldsymbol{f(R,G,B) = min\,(R,G,B)}$$
$$\boldsymbol{delta = max(R,G,B) - min\,(R,G,B)}$$
$$\boldsymbol{V = max\,(R,G,B)}$$

Zasićenost $\boldsymbol{S}$ boje dobiva se tako da se $\boldsymbol{delta}$ podijeli s maksimalnom vrijednosti sjajnosti $\boldsymbol{V}$ od crvene, zelene i plave. Ukoliko je sjajnost $\boldsymbol{V}$ jednaka nuli tada je i zasićenost $\boldsymbol{S}$ jednaka nuli. Kada je zasićenost bojom različita od nule traži se nijansa $\boldsymbol{H}$ boje pomoću izraza i to za svaku od tri boje, crvena, zelena, plava, tako da je odmak za zelenu 120°, za plavu 240° kako je slikovito prikazano na slici 9.

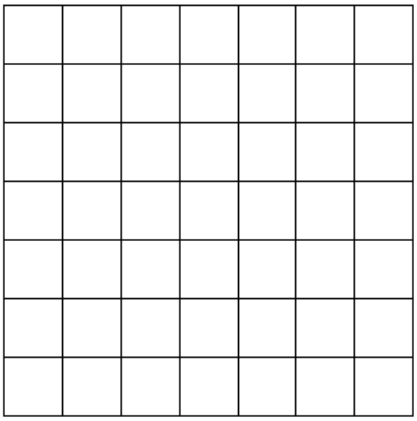
$$\boldsymbol{S = delta/V}$$
$$\boldsymbol{H(R) = 60 \cdot (G - B)/\,delta}$$
$$\boldsymbol{H(G) = 120 + \,60 \cdot (B - R)/\,delta}$$
$$\boldsymbol{H(B) = 240 + \,60 \cdot (R - G)/\,delta}$$

Ovaj matematički zapis preračunavanja iz RGB u HSV je u *Delphi*-u definiran funkcijom RGB2HSV (matricaRGB : TRGB) : THSV;, koja naposljetku samo vraća tri vrijednosti tj. nijansu $\boldsymbol{H}$, zasićenje $\boldsymbol{S}$ i sjajnost $\boldsymbol{V}$:

```
function RGB2HSV (matricaRGB : TRGB) : THSV;
var
  MinRGB, MaxRGB, Delta : Double;
  H , S , V : Double ;
begin
  H := 0.0 ;
  MinRGB := Min (Min( matricaRGB.Red, matricaRGB.Green ), matricaRGB.Blue);
  MaxRGB := Max (Max( matricaRGB.Red, matricaRGB.Green ), matricaRGB.Blue);
  Delta := ( MaxRGB - MinRGB );
  V := MaxRGB ;

If V = 0.0 then
  S := 0.0
else
  S := Delta / V ;

If (S <> 0.0) then
  begin
    If matricaRGB.Red = V then
H := 60.0 * (matricaRGB.Green - matricaRGB.Blue) / Delta
    else If matricaRGB.Green = V then
H := 120.0 + 60.0 * (matricaRGB.Blue - matricaRGB.Red) / Delta
    else
H := 240.0 + 60.0 * (matricaRGB.Red - matricaRGB.Green) / Delta
  end
else
  H := 0;

If H < 0.0 then H := H + 360.0;

If H = 360 then H := 0;

with Result Do
begin
  Hue := round(H) ;          // Hue -> 0..360
  Sat := round(S * 100);     // Saturation -> 0..100 %
  Val := round(V / 2.55);    // Value - > 0..100 %
end;
end;
```

4. FILTER ZA OBRADU SLIKE

U ovom poglavlju će biti izneseni koraci koji su prethodili definiranju filtera kao takvoga. Na početku je definiran zadatak da tražimo objekt određene boje, kojeg korisnik odabere između crvene, zelene i plave. Umjesto piksela uzeti će se polje piksela, polje za promatranje koje će sadržavati 7x7 piksela kako bi se smanjilo vrijeme obrade slike. Na taj način se dobiva smanjenje kvalitete rezolucije slike, ali za ovaj tip zadatka to će biti dovoljno točno za određivanje koordinata objekta na slici kamere.



Slika 10. Polje 7x7 piksela

Odabrano je polje sa neparnim brojem piksela kako bi poslije obrade slike bilo lakše u centru iscrtati pokazivač. Sada kada je definirana veličinu polja, koja može biti proizvoljna – treba voditi računa o tome da se veličinom polja smanjuje mogućnost detekcije objekta što je objekt udaljeniji od fokusa kamere. Potrebno je izračunati ponovo nijansu, zasićenje i sjajnost ali ovoga puta ukupnog polja. Ovdje treba naglasiti kako za nijansu crvene boje imamo pojavu neeuklidske veličine, pošto je raspon crvene boje koji se definira na kružnici HSV sustava postavljen, u ovom rješavanju zadanog zadatka, od 340 do 40 gdje joj vrijednost prelazi preko nule. Stoga će biti upotrijebljena trigonometrija kako bi se riješio problem neeuklidske veličine za nijansu crvene boje. Uvrštavanjem ovoga dijela preračunavanja crvene boje dobivaju se egzaktnije matematičke vrijednosti za crvenu nijansu.

Za određivanje neeuklidske veličine definirana je tablica sa vrijednostima za svaki kut u rasponu od 0° do 360° sa inkrementom za 1°. Sitnija podjela je nepotrebna u ovom radu. U *Delphi-*

u format varijable sa decimalnim mjestima zauzima veliku memoriju te zahtijeva veće vrijeme obrade kada se vrše matematičke operacije nad takvom varijablom. Stoga se ovdje primjenjuje množenje sinusa i kosinusa kuta sa 1000, kako bi koristili zaokružene brojeve, iz razloga što kod računanja arkustangesa tog kuta (koji je ujedno i nijansa crvene boje u HSV sustavu boja) ta se množenja, sinusa i kosinusa sa 1000, dijeljenjem se pokrate. Vektor sa sinusom i kosinusom za vrijednost svakoga brojčano cijelog kuta od 0° do 360° je definirana u *Delphi*-u sintaksom:

```
for i := 0 to 360 do          //Svaki član vektora „tab_Sin_Cos" ima tri podatka u sebi: kut, sinus vrijednost i kosinus vrijednost kuta
  begin
    tab_Sin_Cos[i].Kut := i;                                  //Upis kuta u tablicu
    tab_Sin_Cos[i].Sin_kut := round(Sin(i*Pi/180) * 1000);    //Izračun sinusa za kut i upis u tablicu
    tab_Sin_Cos[i].Cos_Kut := round(Cos(i*Pi/180) * 1000);    //Izračun kosinusa za kut i upis u tablicu
  end;
```

U nastavku je prikazana sintaksa kojom se određuje srednja vrijednost nijanse ovisno o tome da li je riječ o crvenoj ili o druge dvije boje. Ukoliko su zadovoljeni kriteriji da je odabrani referentni uzorak crvene boje tada će se koristeći trigonometrijske funkcije odrediti srednja vrijednost crvene nijanse za referentnu veličinu. Matematički model kod određivanja referentne vrijednosti za crvenu boju:

$$R\_Sin = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7}\sin\left(H_{ji}\right)$$

$$R\_Cos = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7}\cos\left(H_{ji}\right)$$

$$Ref\_HUE = arctg\left(\frac{R\_Sin}{R\_Cos}\right)\cdot\frac{180}{\pi}$$

Za određivanje srednjeg kvadratnog rasipanja oko *Hue* vrijednosti za crvenu nijansu, također se koristi trigonometrija, model je:

$$Uku\_Tan = \sum_{j=1}^{j=7}\sum_{i=1}^{i=7}\left(tg\left(Ref\_HUE\cdot\frac{\pi}{180}\right) - \frac{\sin\left(H_{ji}\right)}{\cos\left(H_{ji}\right)}\right)^2$$

$$Ref\_dHUE = arctg\left(\sqrt{\frac{Uku\_Tan}{49}}\right)\cdot\frac{180}{\pi}$$

Za određivanje srednje vrijednosti nijanse za druge dvije boje i srednje kvadratno rasipanje koristi se aritmetička sredina sviju elemenata u polju:

$$Ref\_HUE = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7} H_{ji}$$

$$Ref\_dHUE = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7}\left|Ref\_HUE\ - H_{ji}\right|$$

Za određivanje srednje vrijednosti zasićenje Ref_SAT i sjajnost Ref_VAL polja koristimo sumu svih piksela u polju te potom dijeljenje sa ukupnim brojem piksela koje sačinjavaju to polje.

$$Ref\_SAT = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7} S_{ji}$$

$$Ref\_VAL = \frac{1}{49}\sum_{j=1}^{j=7}\sum_{i=1}^{i=7} V_{ji}$$

Ref_HUE - referentna vrijednost određena odabirom korisnika koju boju želi pratiti na slici

H_{ji} − vrijednost *Hue* člana matrice sa HSV vrijednostima na adresi *j*,*i*

S_{ji} − vrijednost *Saturation* člana matrice sa HSV vrijednostima na adresi *j*,*i*

V_{ji} − vrijednost *Value* člana matrice sa HSV vrijednostima na adresi *j*,*i*

```
if (ODABRANE_BOJE[1] = TRUE) and (matricaHSV[Y,X].Hue >= 340) or (matricaHSV[Y,X].Hue <= 40)  then
      begin                                                    //Prethodni uvjet zadovoljavaju pikseli crvene nijanse
      R_Sin := 0;
      R_Cos := 0;
        for j := 0 to 6 do
        begin
          y_j := Y - 3 + j;
          for i := 0 to 6 do
            begin
              x_i := X - 3 + i;
              R_Sin := R_Sin + tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Sin_kut;
              R_Cos := R_Cos + tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Cos_kut;
            end;
        end;
        U_HUE := round(arctan(R_Sin / R_Cos)*180 / PI);          //Referentne vrijednosti crvene boje za polje 7x7 piksela
        if U_HUE < 0 then U_HUE := U_HUE + 360;

        Uku_Tan := 0;
        for j := 0 to 6 do
        begin
          y_j := Y - 3 + j;
          for i := 0 to 6 do
            begin
              x_i := X - 3 + i;
              Uku_Tan := Uku_Tan + power((tan(U_HUE/180*PI) - tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Sin_kut /
                         tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Cos_kut), 2);
            end;
        end;
        U_dHUE := round(arctan(power(Uku_Tan / 49, 0.5))*180 / PI);      //Kvadratno rasipanje ref. uzorka crvene boje
      end
    else                                  //Za zelenu i plavu boju određujemo ref. vrijednosti HUE-a i dHUE-a na sljedeći način
      begin
        for j := 0 to 6 do
        begin
          y_j := Y - 3 + j;
          for i := 0 to 6 do
            begin
              x_i := X - 3 + i;
              Hue := Hue + matricaHSV[y_j,x_i].Hue;
            end;
        end;
        U_HUE := round(Hue / 49);
        for j := 0 to 6 do
        begin
          y_j := Y - 3 + j;
          for i := 0 to 6 do
            begin
              x_i := X - 3 + i;
              d_HUE := d_HUE + round(abs(U_HUE - matricaHSV[y_j,x_i].Hue));
            end;
        end;
        U_dHUE := round(d_HUE / 49);
      end;
```

Definiranjem polja koje sadrži informacije o nijansi, zasićenosti i sjajnosti, te rasipanju nijanse, za 7x7 piksela koje treba spremiti, definirana je matrica naziva *matricaHSV_polja7x7* u koju se upisuju vrijednosti pomoću procedure *Robusna_matricaHSV7x7*:

```
procedure Robusna_matricaHSV7x7;
var
  x, y, xa, ya : integer;
  xx, yy, xi, yj : integer;
  kk, ll : integer;
  dH : integer;
begin
  yy := matrica_pretrage.YG + 7;
  while yy <= matrica_pretrage.YD do
   begin
   xx := matrica_pretrage.XL + 7;
   while xx <= matrica_pretrage.XD do
     begin
     grubi_hue := 0;
     grubi_sat := 0;
     grubi_val := 0;
     dH := 0;

     for y := 0 to 6 do
       begin
         yj := yy - 1 - y;
         for x := 0 to 6 do
           begin
               xi := xx - 1 - x;
               grubi_hue := grubi_hue + matricaHSV[yj,xi].Hue;
               grubi_sat := grubi_sat + matricaHSV[yj,xi].Sat;
               grubi_val := grubi_val + matricaHSV[yj,xi].Val;
           end;
       end;

     yj := round((yy)/7-1);
     xi := round((xx)/7-1);
     matricaHSV_polja7x7[yj,xi].Hue := round(grubi_hue/49);
     matricaHSV_polja7x7[yj,xi].Sat := round(grubi_sat/49);
     matricaHSV_polja7x7[yj,xi].Val := round(grubi_val/49);

     for ya := 0 to 6 do
       begin
         kk := yy - 1 - ya;
         for xa := 0 to 6 do
           begin
             ll := xx - 1 - xa;
             dH := dH + abs(matricaHSV_polja7x7[yj,xi].Hue - matricaHSV[kk,ll].Hue);
           end;
       end;

     matricaHSV_polja7x7[yj,xi].dHUE := round(dH/49);

     inc(xx,7);
     end;
   inc(yy,7);
   end;
end;
```

Obrada slike zahtjeva određeno vrijeme za sve kalkulacije, te da bi se smanjilo vrijeme obrade slike postavljene su granice pretrage novih rezultata unutar okvira u kojem se očekuje da će se nalaziti objekt. Zapravo je postavljen malo širi prozor oko samog objekta koji se obrađuje, umjesto da se stalno cijela slika analizira. Ukoliko nestane objekt iz vidnog polja kamere tada se obrađuje cijela slika do ponovne detekcije traženog objekta kada će pretraživanje biti svedeno ponovo na prozor oko objekta prepoznavanja.

4.1. Parametri filtera

Parametri filtera su definirani u *unit*-u *obrada_slike* pod procedurom *obrada(slika_kamere: TBitmap);*. Parametri filtera su podešavani eksperimentalno, nakon svakog postavljanja parametara filtera promatrali su se rezultati. Prvi korak je bio pokušaj da parametri budu jednoznačni za prepoznavanje svih triju boja. Poslije se pokazalo da svaka boja treba drugačije parametre filtera. U trenutku kada korisnik odabere, recimo crvenu boju objekta za praćenje, tada će biti definirani parametri samo za crvenu boju, ukoliko odabere zelenu tada će parametri biti podešeni za zelenu boju. Parametri filetra dodjeljuju se u glavnom *unit*-u – *glavni_prozor*.

Filter sadrži četiri uvjeta. Postavljen je uvjet na zasićenje, sjajnost, nijansu, dok je četvrti uvjet na vrijednost rasipanja nijanse, tako da rasipanje nijanse polja mora biti unutar granica za određenu boju. Uvjeti su napisani u korelaciji apsolutne razlike vrijednosti polja od referentne vrijednosti koja je određena izborom korisnika.

$$|Ref\_Val - matricaHSV\_polja7x7[q,w].Val| \leq Tol\_Val$$

$$|Ref\_Sat - matricaHSV\_polja7x7[q,w].Sat| \leq Tol\_Sat$$

$$|Ref\_Hue - matricaHSV\_polja7x7[q,w].Hue| \leq Tol\_Hue$$

$$|Ref\_dHue - matricaHSV\_polja7x7[q,w].dHue| \leq Tol\_dHue$$

Ukoliko polje zadovoljava ova četiri uvjeta to polje još dobiva *Boolean*-ov argument *Istinit* (eng. *True*). Nakon prolaska slike kroz filter dobijemo granice unutar kojih se nalazi traženi objekt. Ova obrada je primijenjena na predmete ravne plohe pravokutna i okruglog oblika. Sa zakrivljenim površinama uvjeti osvjetljenja okoline dosta mijenjuju karakter podloge u boji koja se obrađuje pa se dobije podrhtavajući pokazivač. Sintaksa filtera:

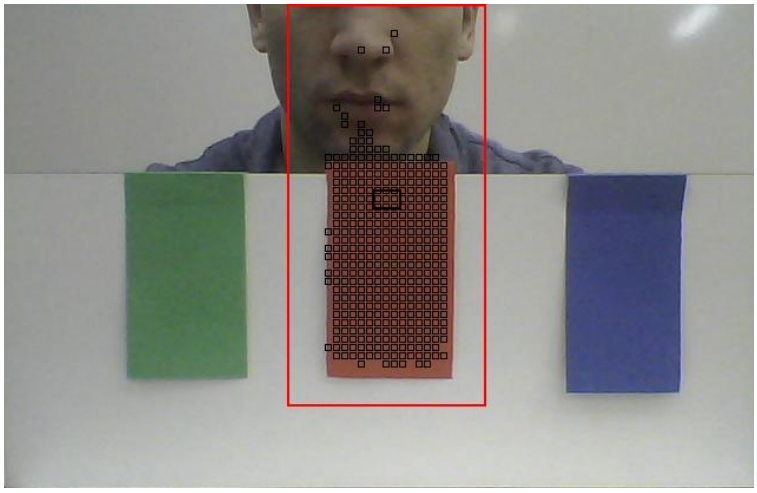
```
procedure obrada(slika_kamere: TBitmap);
var
  q, w : integer;
  rac_q, rac_w : integer;
begin

    razmak_grupe := 21;
    razmak_pretrage := 35;
    razmak_razlike := razmak_pretrage - razmak_grupe;

    if prvi_prolaz = true then
      begin
         XL_provjere := round(prozor_provjere[1]/7);
         XD_provjere := round(prozor_provjere[2]/7);
         YG_provjere := round(prozor_provjere[3]/7);
         YD_provjere := round(prozor_provjere[4]/7);
         matrica_pretrage.XL := prozor_provjere[1];
         matrica_pretrage.XD := prozor_provjere[2];
         matrica_pretrage.YG := prozor_provjere[3];
         matrica_pretrage.YD := prozor_provjere[4];
      end
    else
      begin
         XL_provjere := 0;
         XD_provjere := 90;
         YG_provjere := 0;
         YD_provjere := 67;
         matrica_pretrage.XL := 1;
         matrica_pretrage.XD := 638;
         matrica_pretrage.YG := 2;
         matrica_pretrage.YD := 478;
      end;

slika_u_matricuRGB(slika_kamere);

if (Odabrana_boja <> 0) then
  begin
    novi_XL_pretrage := 638;
    novi_XD_pretrage := 1;
    novi_YG_pretrage := 478;
    novi_YD_pretrage := 2;

    novi_XL_grupe := 638;
    novi_XD_grupe := 1;
    novi_YG_grupe := 478;
    novi_YD_grupe := 2;

    i := 0;

    for q := YG_provjere to YD_provjere do
    begin
      rac_q := q*7 + 2 - 3;
      for w := XL_provjere to XD_provjere do
      begin
        rac_w := w*7 + 1 - 3;
        if (abs(Ref_Val - matricaHSV_polja7x7[q,w].Val) <= Tol_VAL) and
           (abs(Ref_Sat - matricaHSV_polja7x7[q,w].Sat) <= Tol_SAT) and
           (abs(Ref_dHUE - matricaHSV_polja7x7[q,w].dHUE) <= Tol_dHUE) and
           (abs(Ref_HUE - matricaHSV_polja7x7[q,w].Hue) <= Tol_HUE) then
            begin                    //Određivanje granica slike unutar kojega se nalaze objekti odabrane boje (nijanse)
              matricaHSV_polja7x7[q,w].Indikator_boje := true;
              if novi_XL_pretrage > rac_w then novi_XL_pretrage := rac_w;
              if novi_XD_pretrage < rac_w then novi_XD_pretrage := rac_w;
              if novi_YG_pretrage > rac_q then novi_YG_pretrage := rac_q;
              if novi_YD_pretrage < rac_q then novi_YD_pretrage := rac_q;
            end
          else
             matricaHSV_polja7x7[q,w].Indikator_boje := false;
      end;
    end;
```

```
for q := YG_provjere to YD_provjere do                          //Grupiranje polja koja odgovaraju referentnoj vrijednosti
  for w := XL_provjere to XD_provjere do
     if matricaHSV_polja7x7[q,w].Indikator_boje then
        begin
          if (matricaHSV_polja7x7[q-1,w-1].Indikator_boje = false) and
             (matricaHSV_polja7x7[q-1,w].Indikator_boje = false) and
             (matricaHSV_polja7x7[q-1,w+1].Indikator_boje = false) and
             (matricaHSV_polja7x7[q,w-1].Indikator_boje = false) then
              begin
                i := i + 1;
                matricaHSV_polja7x7[q,w].Broj_grupe := i;
              end
          else
               if matricaHSV_polja7x7[q-1,w-1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w-1].Broj_grupe
               else if matricaHSV_polja7x7[q-1,w].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w].Broj_grupe
               else if matricaHSV_polja7x7[q-1,w+1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w+1].Broj_grupe
               else if matricaHSV_polja7x7[q,w-1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w-1].Broj_grupe;
        end
     else
       matricaHSV_polja7x7[q,w].Broj_grupe := 0;

for q := YD_provjere downto YG_provjere do                      //Grupiranje polja koja odgovaraju istoj grupi
  for w := XD_provjere downto XL_provjere do
     if matricaHSV_polja7x7[q,w].Broj_grupe <> 0 then
        begin
               if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w+1].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w+1].Broj_grupe <> 0) then
                         matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w+1].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w].Broj_grupe <> 0) then
                         matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w-1].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w-1].Broj_grupe <> 0) then
                         matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w-1].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q,w+1].Broj_grupe) and
                    (matricaHSV_polja7x7[q,w+1].Broj_grupe <> 0) then
                         matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w+1].Broj_grupe
        end;

Odabrana_Grupa :=  matricaHSV_polja7x7[round(Y_koord_pokazivaca / 7),round(X_koord_pokazivaca / 7)].Broj_grupe;

for q := YG_provjere to YD_provjere do          //Određivanje granica grupe objekta kojega je korisnik odabrao
  begin
    rac_q := q*7 + 2 - 3;
    for w := XL_provjere to XD_provjere do
      begin
        rac_w := w*7 + 1 - 3;
        if matricaHSV_polja7x7[q,w].Broj_grupe = Odabrana_Grupa then
           begin
             if novi_XL_grupe > rac_w then novi_XL_grupe := rac_w;
             if novi_XD_grupe < rac_w then novi_XD_grupe := rac_w;
             if novi_YG_grupe > rac_q then novi_YG_grupe := rac_q;
             if novi_YD_grupe < rac_q then novi_YD_grupe := rac_q;
           end;
      end;
  end;

 X_koord_pokazivaca := round((novi_XL_grupe + novi_XD_grupe)/2);
 Y_koord_pokazivaca := round((novi_YG_grupe + novi_YD_grupe)/2);
```

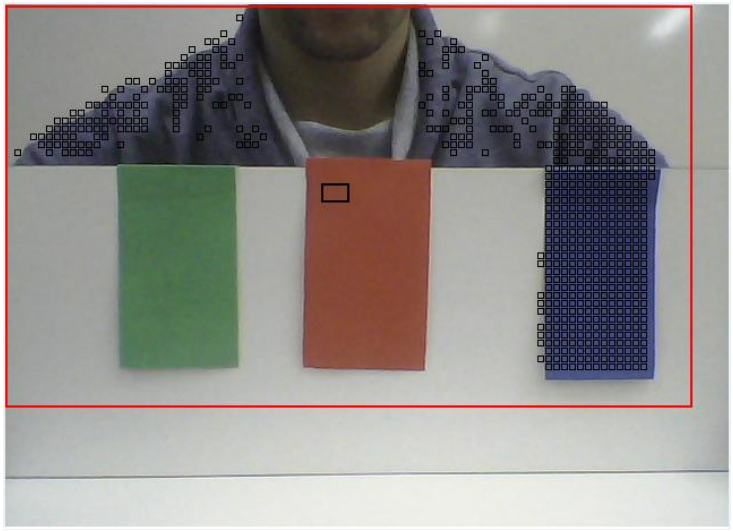
```
    if (novi_XL_pretrage - razmak_pretrage) < 1 then prozor_provjere[1] := 1
      else  prozor_provjere[1] := novi_XL_pretrage - razmak_pretrage;
    if (novi_XD_pretrage + razmak_pretrage) > 638 then prozor_provjere[2] := 638
      else  prozor_provjere[2] := novi_XD_pretrage + razmak_pretrage;
    if (novi_YG_pretrage - razmak_pretrage) < 2 then prozor_provjere[3] := 2
      else  prozor_provjere[3] := novi_YG_pretrage - razmak_pretrage;
    if (novi_YD_pretrage + razmak_pretrage) > 478 then prozor_provjere[4] := 478
      else  prozor_provjere[4] := novi_YD_pretrage + razmak_pretrage;

    if (novi_XL_grupe - razmak_grupe) < 1 then novi_XL_grupe := 1
      else  novi_XL_grupe := novi_XL_grupe - razmak_grupe;
    if (novi_XD_grupe + razmak_grupe) > 638 then novi_XD_grupe := 638
      else  novi_XD_grupe := novi_XD_grupe + razmak_grupe;
    if (novi_YG_grupe - razmak_grupe) < 2 then novi_YG_grupe := 2
      else  novi_YG_grupe := novi_YG_grupe - razmak_grupe;
    if (novi_YD_grupe + razmak_grupe) > 478 then novi_YD_grupe := 478
      else  novi_YD_grupe := novi_YD_grupe + razmak_grupe;

  if Odabrana_Grupa = 0 then
  begin

    prozor_provjere[1] := 1 + razmak_razlike;
    prozor_provjere[2] := 638 - razmak_razlike;
    prozor_provjere[3] := 2 + razmak_razlike;
    prozor_provjere[4] := 478 - razmak_razlike;

    novi_XL_grupe := 1 + 2*razmak_razlike;
    novi_XD_grupe := 638 - 2*razmak_razlike;
    novi_YG_grupe := 2 + 2*razmak_razlike;
    novi_YD_grupe := 478 - 2*razmak_razlike;

  end;

  for q := 0 to 67 do
    for w := 0 to 90 do
       if matricaHSV_polja7x7[q,w].Broj_grupe = Odabrana_Grupa then  brojac := brojac + 1;

 velicina := brojac;

 if (velicina < 9) then
    prvi_prolaz := false
 else
    prvi_prolaz := true;

  brojac := 0;
end;

end;
```

Nakon ove prethodne procedure dobiva se nekoliko informacija: saznajemo koordinate težišta objekta o boji koju želimo pratiti, provjeravamo veličinu prepoznatog objekta te ako je manja od 75x75 piksela ne prikazuje se pokazivač već se uzima novu sliku na obradu, ukoliko je pronađen objekt određene boje u slici kamere tada definiramo *prozor provjere* koji služi da u ponovnom obrađivanju slike traži se objekt unutar *prozora provjere* kako bi se skratilo vrijeme obrade slike, tj. reducirala količina posla na dijelovima slike gdje nema objekta kao takvog. Postavljeno je da se obrada slike izvršava svakih 200 ms.

4.2. Rezultati obrade slike bez grupiranja polja svakog objekta



Slika 11. Parametri filtera za crveni uzorak su nedovoljno podešeni za isključivanje šuma



Slika 12. Parametri filtera za plavi uzorak su nedovoljno podešeni za isključivanje šuma

Slika 11. prikazuje kako parametri filtera nisu dovoljno točno podešeni za crveni uzorak boje te tako kroz filter prolazi šum crvenog pigmenta (šum na slici predstavlja lice koje se nalazi pred web-kamerom). Parametri ovoga filtera su:

$$|Ref\_Hue - matricaHSV\_polja7x7[j,i].Hue| \leq 15$$

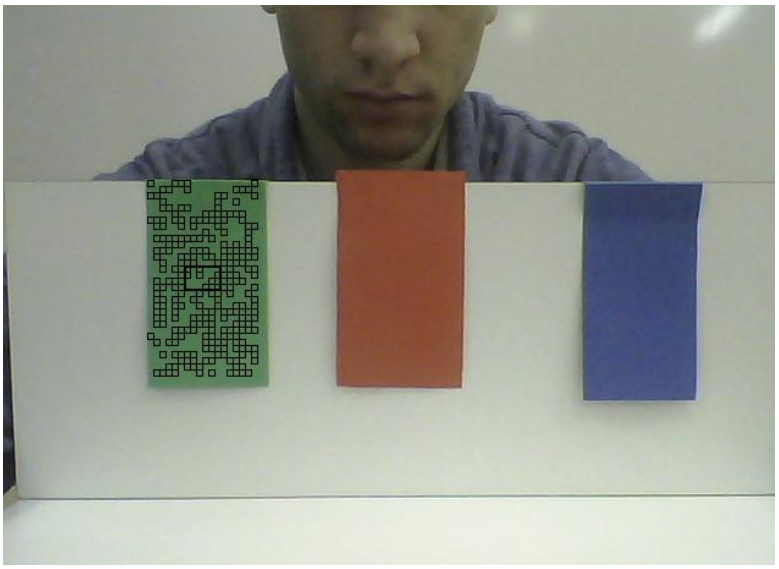
$$|Ref\_Sat - matricaHSV\_polja7x7[j,i].Sat| \leq 30$$

$$|Ref\_Val - matricaHSV\_polja7x7[j,i].Val| \leq 30$$

$$|Ref\_dHue - matricaHSV\_polja7x7[j,i].dHue| \leq 10$$

Nakon izmijene parametara filtera dobiju se rezultati na Slici 12. koji ne daju dobre rezultate prepoznavanja za plavi uzorak. Majica osobe ispred kamere je uzrokovala veliki šum za plavi uzorak. Crveni okvir na slici pokazuje veličinu prozora provjere, tj. koji dio slike se obrađuje. Parametri ovoga filtera su:

$$|Ref\_Hue - matricaHSV\_polja7x7[j,i].Hue| \leq 10$$
$$|Ref\_Sat - matricaHSV\_polja7x7[j,i].Sat| \leq 20$$
$$|Ref\_Val - matricaHSV\_polja7x7[j,i].Val| \leq 20$$
$$|Ref\_dHue - matricaHSV\_polja7x7[j,i].dHue| \leq 10$$



Slika 13. Parametri filtera za zeleni uzorak su dovoljno dobro podešeni za isključivanje šuma

Na slici 13. vidimo isključivo prepoznavanje zelenog uzorka bez šuma, međutim prikaz prepoznatih polja koje zadovoljavaju parametre ovog filtera je jako loš te je prepoznati uzorak nekompaktan. U realnom vremenu, u ovom slučaju detektira se da polje u jednom času može biti prepoznato, a u drugome ne, stoga ovaj prikaz dosta treperi s obzirom na prikaz ćelija koje zadovoljavaju ovaj filter. U ovom filteru je samo jedan uvjet s kojim se uspjelo ugasiti šum ali ne i zadovoljiti kriterij kompaktnosti prepoznavanja objekta zelene boje.

$$|Ref\_Hue - matricaHSV\_polja7x7[j,i].Hue| \leq 15$$

Slike 14., 15., 16. prikazuju rezultate filtera za crvenu, zelenu i plavu boju za koje su parametri filtera zasebno podešeni, a oblik uvjeta filtera je jednako definiran. To znači da svaka boja ima svoje granice tolerancije na svaki uvjet koji je jednoznačno definiran za sve tri boje.

$$|Ref\_Hue - matricaHSV\_polja7x7[j,i].Hue| \leq Tol\_HUE$$

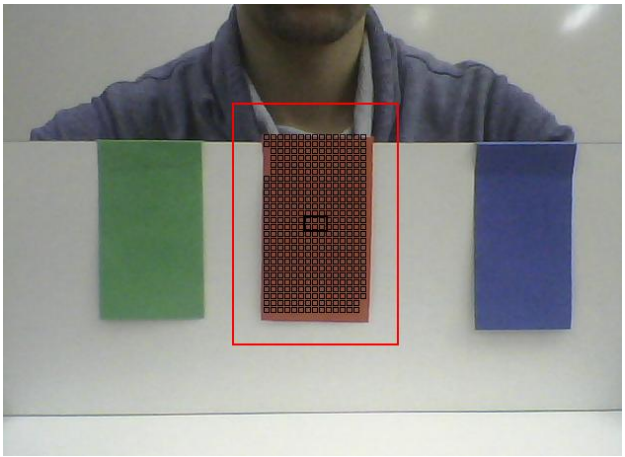
$$|Ref\_Sat - matricaHSV\_polja7x7[j,i].Sat| \leq Tol\_SAT$$

$$|Ref\_Val - matricaHSV\_polja7x7[j,i].Val| \leq Tol\_VAL$$

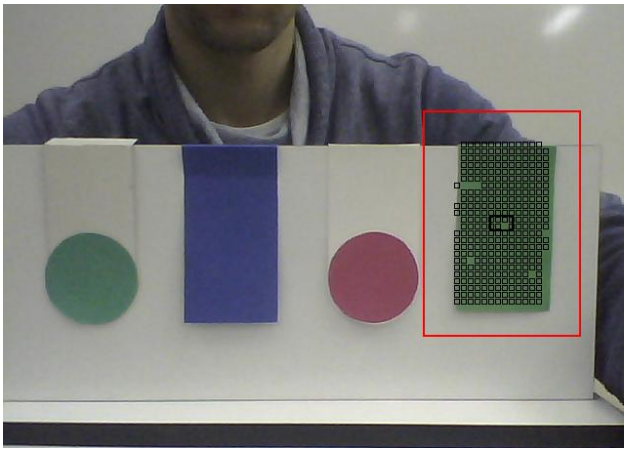
$$|Ref\_dHue - matricaHSV\_polja7x7[j,i].dHue| \leq Tol\_dHUE$$

Tablica 1. Tolerancijske vrijednosti na parametre filtera

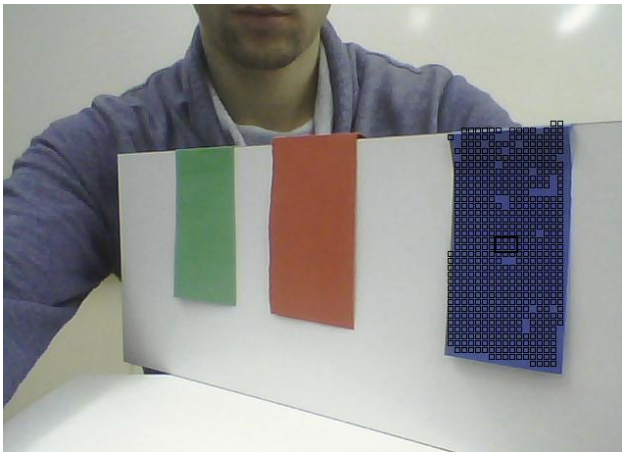
	CRVENA	ZELENA	PLAVA
Tol_HUE	8	15	5
Tol_SAT	15	15	10
Tol_VAL	20	20	20
Tol_dHUE	5	5	5



Slika 14. Rezultati filtera za crveni objekt

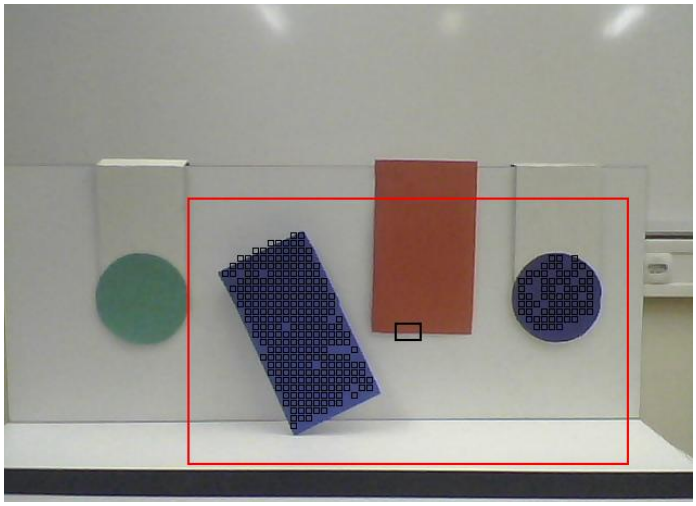


Slika 15. Rezultati filtera za zeleni objekt

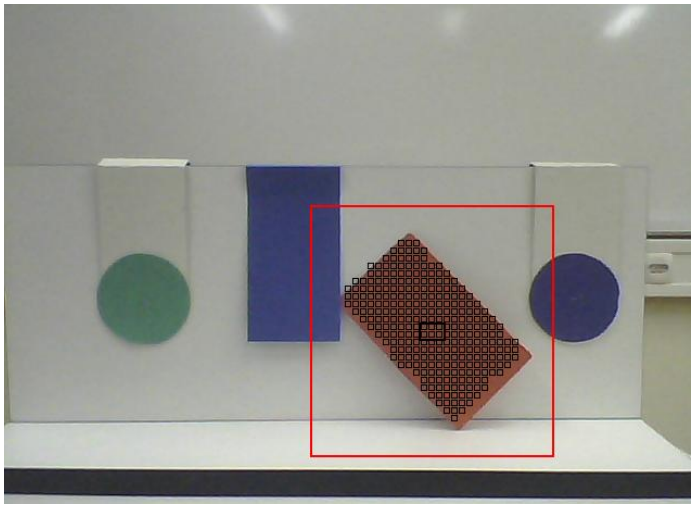


Slika 16. Rezultati filtera za plavi objekt

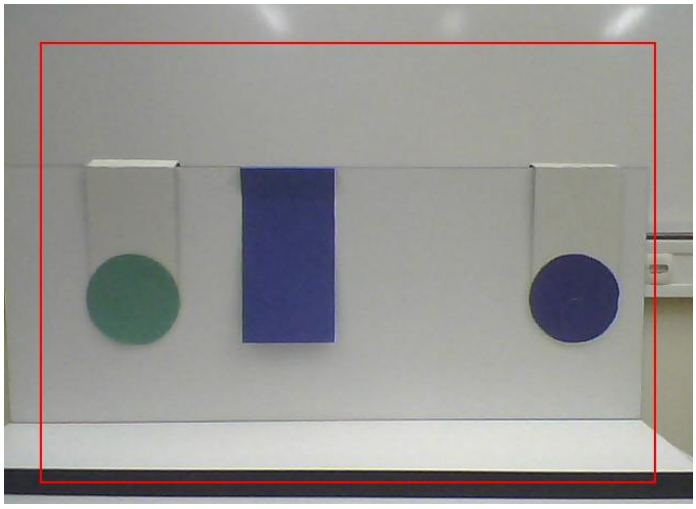
Slika 17. zorno prikazuje problem koji nastaje kada se nađu pred kamerom dva objekta iste boje, te kako ovakva vrsta filtera nije dovoljna već je potrebno izvršiti grupiranje prepoznatih polja u klase. Potom davajući vrijednost klasama odrediti koji objekt da se prati. Pokazivač je mali pravokutnik zadebljanih bridova što se nalazi između dva objekta plave boje. Crveni okvir opisuje prozor obrade slike. Slika 18. i 19. prikazuju da se pokazivač smješta u težište objekta, te koliki je prozor provjere kada objekt praćenja nestane iz vidnog polja kamere (Slika 19.).



Slika 17. Dva objekta iste boje pred kamerom



Slika 18. Pokazivač smješten u težište objekta na slici



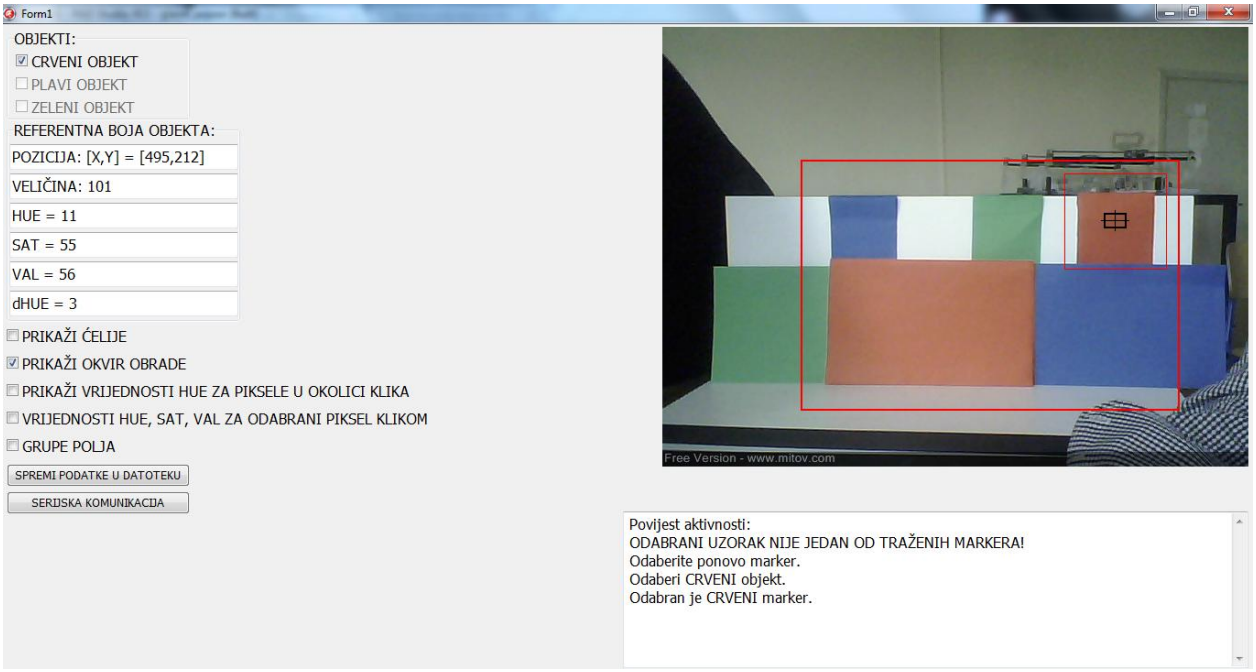
Slika 19. Prozor provjere kada nema ciljanog objekta pred kamerom (usporedno sa Slikom 18.)

4.3. Grupiranje polja kad je više objekata iste boje na slici

U poglavlju 4.1. Parametri filtera, u sintaksi programa je prikazana metoda kojom su izvršava grupiranje objekata kojih može biti više na slici, a da su iste boje. Sintaksa:

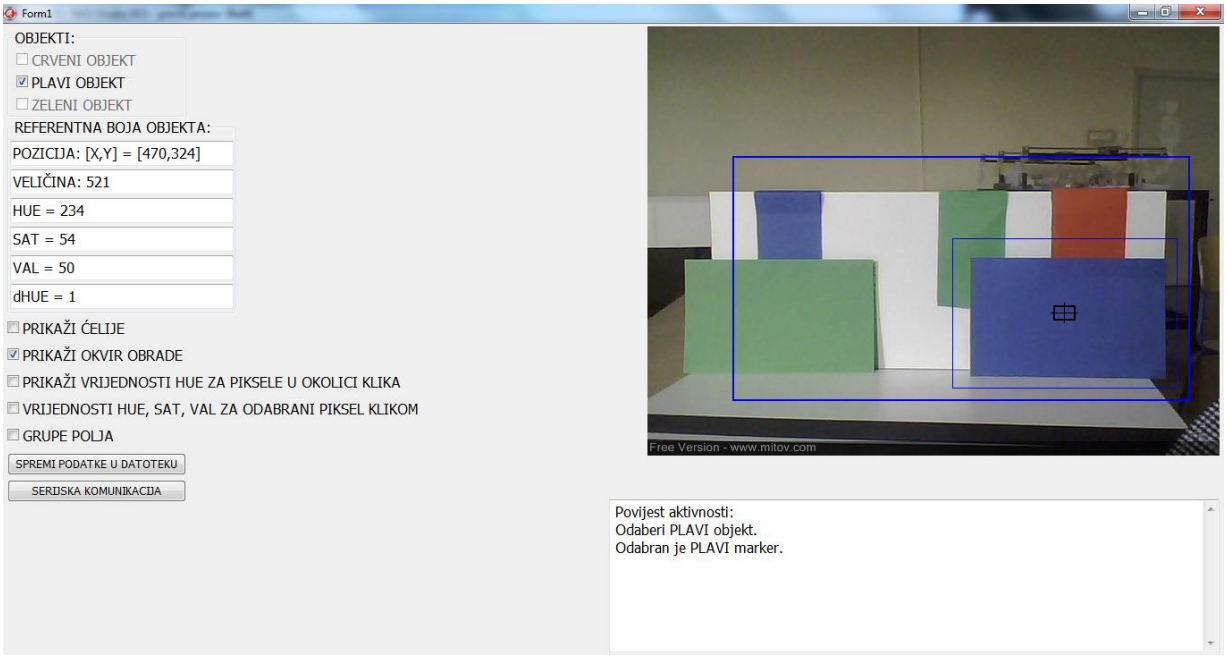
```
for q := YG_provjere to YD_provjere do                          //Grupiranje polja koja odgovaraju referentnoj vrijednosti
  for w := XL_provjere to XD_provjere do
     if matricaHSV_polja7x7[q,w].Indikator_boje then
        begin
          if (matricaHSV_polja7x7[q-1,w-1].Indikator_boje = false) and
             (matricaHSV_polja7x7[q-1,w].Indikator_boje = false) and
             (matricaHSV_polja7x7[q-1,w+1].Indikator_boje = false) and
             (matricaHSV_polja7x7[q,w-1].Indikator_boje = false) then
              begin
                i := i + 1;
                matricaHSV_polja7x7[q,w].Broj_grupe := i;
              end
          else
               if matricaHSV_polja7x7[q-1,w-1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w-1].Broj_grupe
               else if matricaHSV_polja7x7[q-1,w].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w].Broj_grupe
               else if matricaHSV_polja7x7[q-1,w+1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w+1].Broj_grupe
               else if matricaHSV_polja7x7[q,w-1].Indikator_boje then
                    matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w-1].Broj_grupe;
        end
     else
       matricaHSV_polja7x7[q,w].Broj_grupe := 0;

for q := YD_provjere downto YG_provjere do    //Grupiranje polja koja odgovaraju istoj grupi
  for w := XD_provjere downto XL_provjere do
     if matricaHSV_polja7x7[q,w].Broj_grupe <> 0 then
        begin
               if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w+1].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w+1].Broj_grupe <> 0) then
                              matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w+1].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w].Broj_grupe <> 0) then
                              matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w-1].Broj_grupe) and
                    (matricaHSV_polja7x7[q+1,w-1].Broj_grupe <> 0) then
                              matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w-1].Broj_grupe
               else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q,w+1].Broj_grupe) and
                    (matricaHSV_polja7x7[q,w+1].Broj_grupe <> 0) then
                              matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w+1].Broj_grupe
        end;

Odabrana_Grupa :=  matricaHSV_polja7x7[round(Y_koord_pokazivaca / 7),round(X_koord_pokazivaca / 7)].Broj_grupe;

for q := YG_provjere to YD_provjere do          //Određivanje granica grupe objekta kojega je korisnik odabrao
  begin
    rac_q := q*7 + 2 - 3;
    for w := XL_provjere to XD_provjere do
      begin
        rac_w := w*7 + 1 - 3;
        if matricaHSV_polja7x7[q,w].Broj_grupe = Odabrana_Grupa then
           begin
             if novi_XL_grupe > rac_w then novi_XL_grupe := rac_w;
             if novi_XD_grupe < rac_w then novi_XD_grupe := rac_w;
             if novi_YG_grupe > rac_q then novi_YG_grupe := rac_q;
             if novi_YD_grupe < rac_q then novi_YD_grupe := rac_q;
           end;
      end;
  end;
```

U nastavku prikazivanja rezultata koristio sam dvije informacije: jedna informacija pokazuje koliki je prozor unutar slike u kojem se nalaze svi objekti iste boje, druga informacija jest prozor unutar prethodnog prozora koji daje informaciju samo o onom objektu kojega je korisnik odabrao da ga se prati. Rezultate su prikazani pomoć kvadrata čiji rub ima odgovarajuću boju objekta kojega se prati. Rezultati:



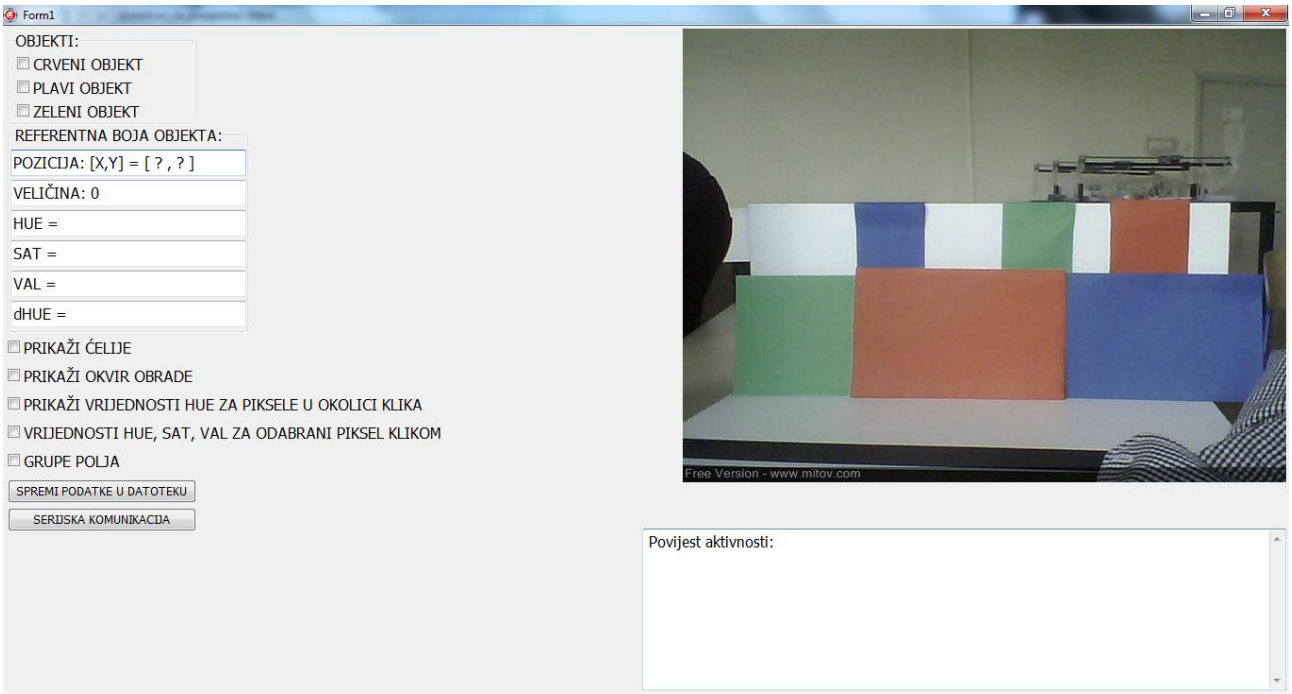
Slika 20. Grupiranje polja kad ima više objekata crvene boje na slici

Slika 21. Grupiranje polja kad ima više objekata zelene boje na slici



Slika 22. Grupiranje polja kad ima više objekata plave boje na slici

5. SUČELJE U *DELPHI*-U



Slika 23. Korisničko sučelje u *Delphi*-u za prepoznavanje objekata prema boji

Korisničko sučelje čini glavni prozor *Form*-e koje je oblikovano tako da označavanjem nekog objekta u *Form*-i ono postaje aktivno. U lijevom gornjem kutu su postavljena dva *groupbox*-a nazvana: *OBJEKTI* i *REFERENTNA BOJA OBJEKTA*. Korisnik odabirom unutar *groupbox*-a *OBJEKTI* odabire boju objekta (*checkbox*) koju želi pratiti. Zatim klikom miša na slici izvrši odabir vidljivog objekta pred kamerom. Ispod slike se nalazi jedan *Memo* prozor preko kojega korisnik dobiva upute što treba napraviti. Kod samog odabira objekta na slici, prema prethodnom odabiru objekta pomoću *checkbox*-a, kojega želimo pratiti, mogu se pojaviti dvije povratne informacije. Prva informacija nas obavještava ukoliko nije odabran objekt nijanse ispred koje je potvrđena kučica (Slika 20., 21. i 22.) da uzorak sa slike nije odgovarajući prema prethodno definiranom izboru objekta kojega želimo pratiti. Drugo upozorenje nam se može pojaviti ukoliko odabirom točne boje nemamo dovoljno informacija za taj objekt. Razlozi tome mogu biti nedovoljna osvijetljenosti ili pak slučaj sjene preko odabranog objekta. Tada je potrebno podesiti vanjske uvjete osvjetljenja ili klikom miša samo odabrati neki drugi dio objekta na slici kako bi bio prepoznat od strane programa.

U drugome *groupbox*-u se nalaze podaci o samome uzorku koji je korisnik odabrao. Prozor *REFERENTNA BOJA OBJEKTA* sadrži informacije položaja objekta koji je sveden na koordinate pokazivača koji se nalazi na slici, veličini objekta (veličinu određuje broj polja od 7x7 piksela koji su prepoznati za odabrani objekt na slici). Ispod su navedene tri informacije o referentnom uzorku kao što su *HUE* (hrv. nijansa), *SAT* (hrv. zasićenje) te *VAL* (hrv. sjajnost).

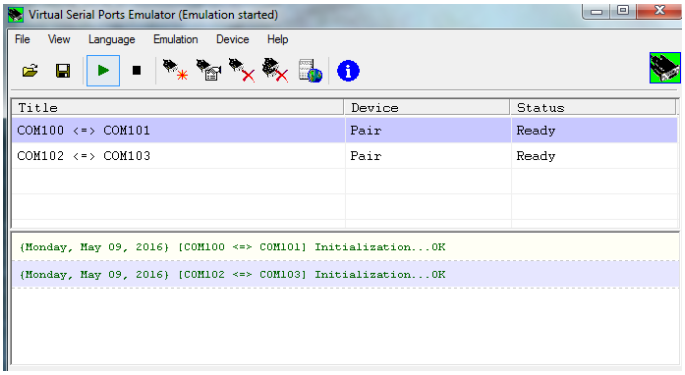
Potom su postavljene opcije ukoliko se želi vidjeti prikaz ćelija koje sačinjavaju objekt (prethodne slike 14., 15., 16., preko objekta koji je prepoznat pokazuju se iscrtani kvadratići). Sljedeća je opcija prikaz okvira pretrage i okvira grupe objekta (slika 20., 21., 22.) unutar kojega se vrši pretraživanje odabrane boje objekta te grupe. „*PRIKAŽI VRIJEDNOSTI HUE ZA PIKSELE U OKOLICI KLIKA"* je izbor kada želimo vidjeti rasprostranjenost vrijednosti nijanse oko piksela na koji smo kliknuli mišem. Također se na ovaj odabir prikazuju uprosječene vrijednosti *HUE* za polja (sačinjena od piksela 7x7) oko piksela na koji se klikne mišem u slici. Četvrti izbor vraća povratnu informaciju za mjesto klika mišem tako što nam u prozor *Povijest aktivnosti* ispisuje *hue, saturation* i *value* za odabrani piksel. Peti izbor omogućava prikaz objekata prema grupi kojoj pripadaju, u ovom slučaju otvara se *Memo* prozor gdje se prikazuju vrijednosti grupa u koju spadaju objekti na slici.

U dnu imamo dva gumba, s prvim pohranjujemo sliku u memoriju u *.bmp* formatu, binarni zapis slike i grupirana polja. Sa drugim gumbom definira se *Port* preko kojega će se vršiti serijska komunikacija sa drugom aplikacijom. Serijska komunikacija se koristi za slanje podataka o objektu – koordinate, veličinu i boju objekta.

Cijela sintaksa programa je podijeljena u tri cjeline, unit-a. Cjeline se zovu *glavni_prozor, obrada_slike, slika_u_matrici*. U glavnom unit-u *glavni_prozor* je opisano grafičko sučelje koje je na slici 23.. U unit-u *slika_u_matrici* memorira se slika sa kamere u matrice sa RGB vrijednostima, u matricu sa HSV vrijednostima, u treću matricu je upisana vrijednost nijanse, zasićenja i sjajnosti polja, koja su sačinjena od skupa piksela 7x7, te je opisana funkcija za transformaciju boje iz RGB-a u HSV. Treći unit *obrada_slike* sadržava parametre filtera slike u kojem se također određuju granice ponovnog pretraživanja slike te određuje veličina objkta koja je prepoznata na slici.

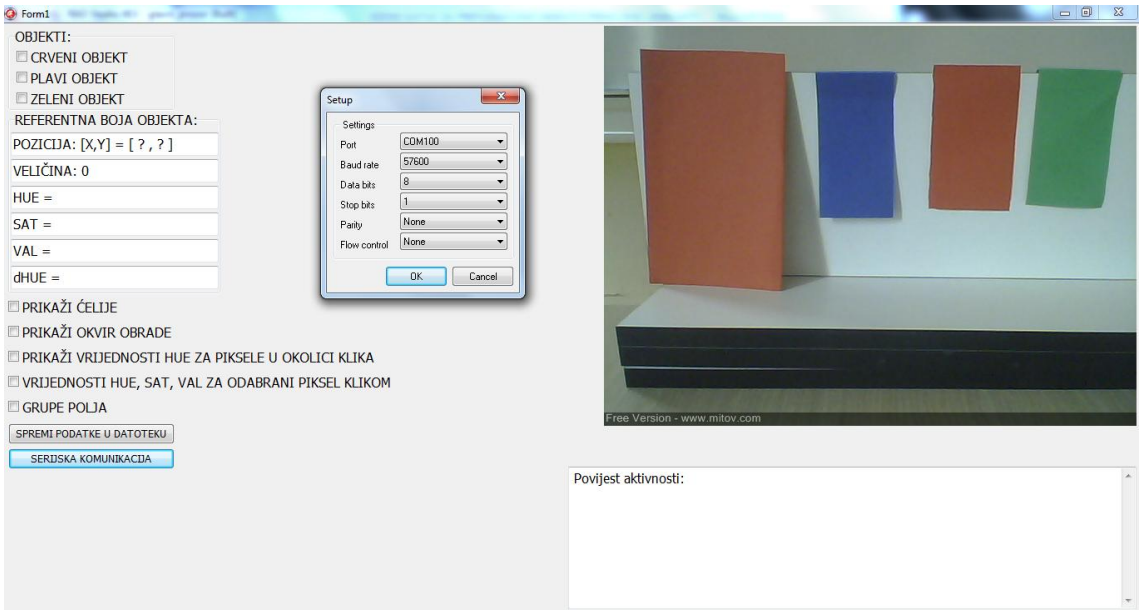
5.1. Serijska komunikacija

Za komunikaciju se koristi virtualni serijski simulator. VSPE (eng. *Virtual Serial Ports Emulator*) – koji služi za serijsku komunikaciju sa drugom aplikacijom. Potrebno je definirati protokol kojim će se slati podatke, o objektu na slici s web-kamere mobilnog vozila, u cilju uspješne navigacije. Ovdje se koristi simulator protokola za serijsku komunikaciju preko kojega se uspostavlja, tj. odrede ulazi (eng. *Port*-ove) za serijsku komunikaciju. Aplikacija za prepoznavanje objekta prema boji ima *Port* 100, dok aplikacija zvana *MyTerminal* ima prijemski *Port* 101. Preko VSPE su definirana ova dva *port*-a te su dodijeljena aplikaciji za praćenje i terminalu koji će ispisivati podatke. Slika 24.



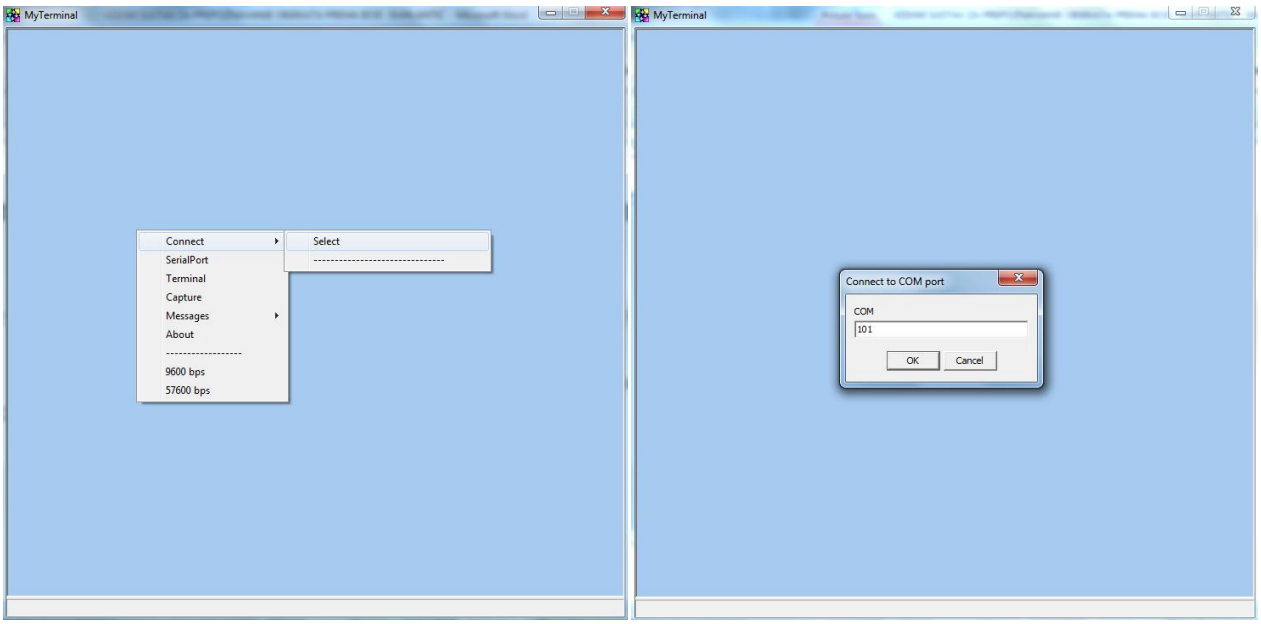
Slika 24. Okruženje VSPE-a

U *Delphi*-u pomoću gumba „SERIJSKA KOMUNIKACIJA“ otvaramo izbornik u kojem odabiremo *Port* 100 i brzinu komunikacije (*Baud rate*) 57600 bps. Slika 25.



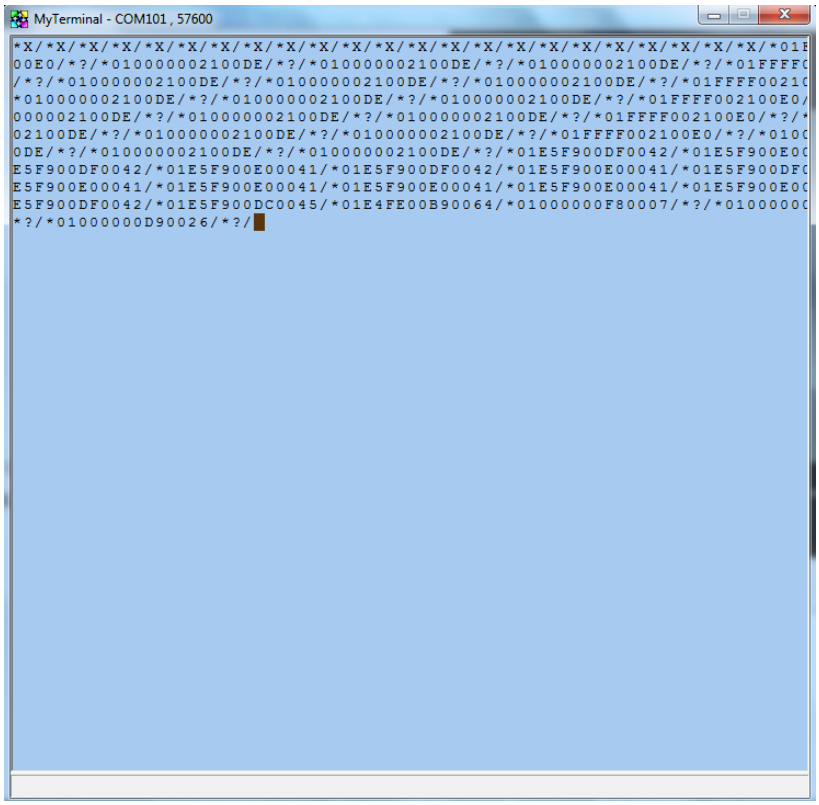
Slika 25. Otvoren izbornik za određivanje serijske veze

U aplikaciji *MyTerminal* također definiramo *Port* 101 i brzinu komunikacije 57600 bps, Slika 26.



Slika 26. Određivanje *Port*-a 101 na MyTerminal-u

Prilikom komunikacije šalju se podaci o boji, položaju objekta po x osi, položaju po y osi, zatim podaci o veličini objekta na slici. Posljednje dvije znamenke su kontrolne, tzv. *Checksum*. Svaki zapis počinje zvjezdicom „*“ i završava kosom crtom „/“. Između ta dva znaka zapis se čita dvije po dvije znamenke. Prve dvije znamenke sadrže boju objekta na slici, druge dvije su koordinate po x-osi, sljedeće dvije varijable su koordinate po y-osi, zatim dolaze dvije nule „00“ (parametri koje ne koristimo u ovoj komunikaciji), potom dvije varijable koje definiraju veličinu objekta prepoznatog na slici, zatim opet dvije nule „00“ te naposljetku dvije varijable koje definiraju *Checksum*. Serijskom vezom zapisi se prenose u heksadekadskom formatu. Na slici 27. prikazane su u *MyTerminal*-u poruke koje su zaprimljene iz aplikacije prepoznavanja objekata prema boji. Komunikacijom se još šalju dva znaka, „X“ i „?“. Znak „X“ šalje poruku da nije odabran niti jedan objekt određene boje koji bi se trebalo prepoznavati sa slike, dok znak „?“ ukazuje da nema traženog objekta u slici.



Slika 27. Poruke zaprimljene serijskom komunikacijom

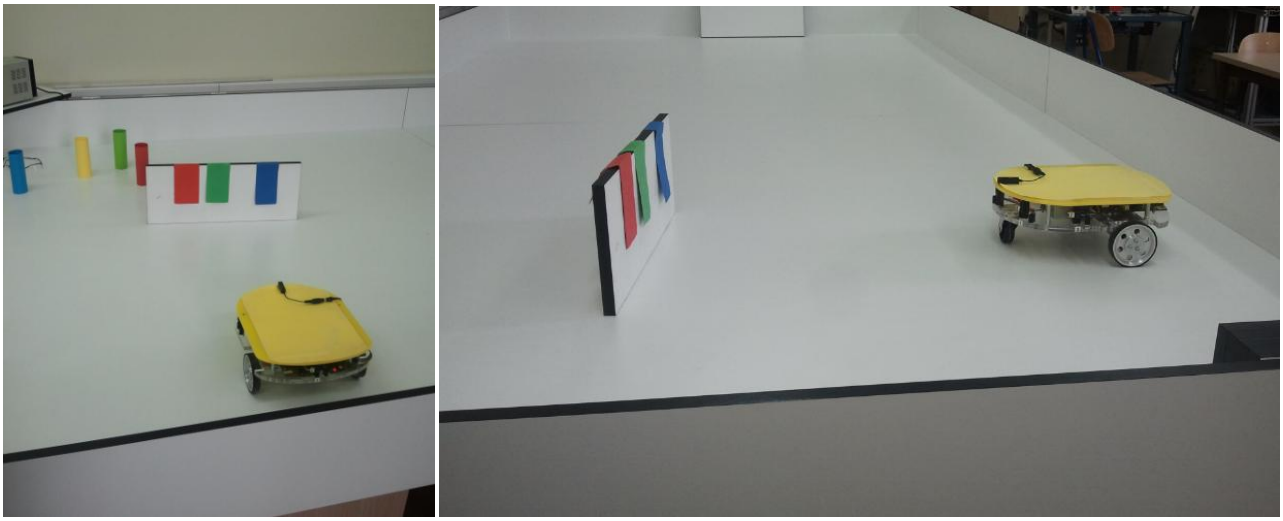
Sintaksa zapisana u *Delphi*-u za serijsku komunikaciju je:

```
procedure TForm1.KOMUNIKACIJATimer(Sender: TObject);
var
  bb, xx, yy, nn, CS : Byte;
  bbxxyy00nn00CS : string;
  razlika : Byte;
begin
  if Odabrana_boja <> 0 then
    begin
      if Odabrana_boja = 1 then bb := 1
        else if Odabrana_boja = 2 then bb := 6
          else bb := 4;

      if X_koord_pokazivaca >= centar_X then
        xx := round((X_koord_pokazivaca - centar_X) / 8)
      else
        xx := 255 + round((X_koord_pokazivaca - centar_X) / 8) + 1;

      if Y_koord_pokazivaca >= centar_Y then
        yy := round((Y_koord_pokazivaca - centar_Y) / 8)
      else
        yy := 255 + round((Y_koord_pokazivaca - centar_Y) / 8) + 1;

      if velicina <= 127 then
        nn := velicina
      else
        nn := round(velicina / 9) + 113;

      CS := Lo(256 - Lo(bb + xx + yy + nn));

     bbxxyy00nn00CS := IntToHex(bb,2) + IntToHex(xx,2) + IntToHex(yy,2) + '00' +  IntToHex(nn,2) + '00' + IntToHex(CS,2);
```

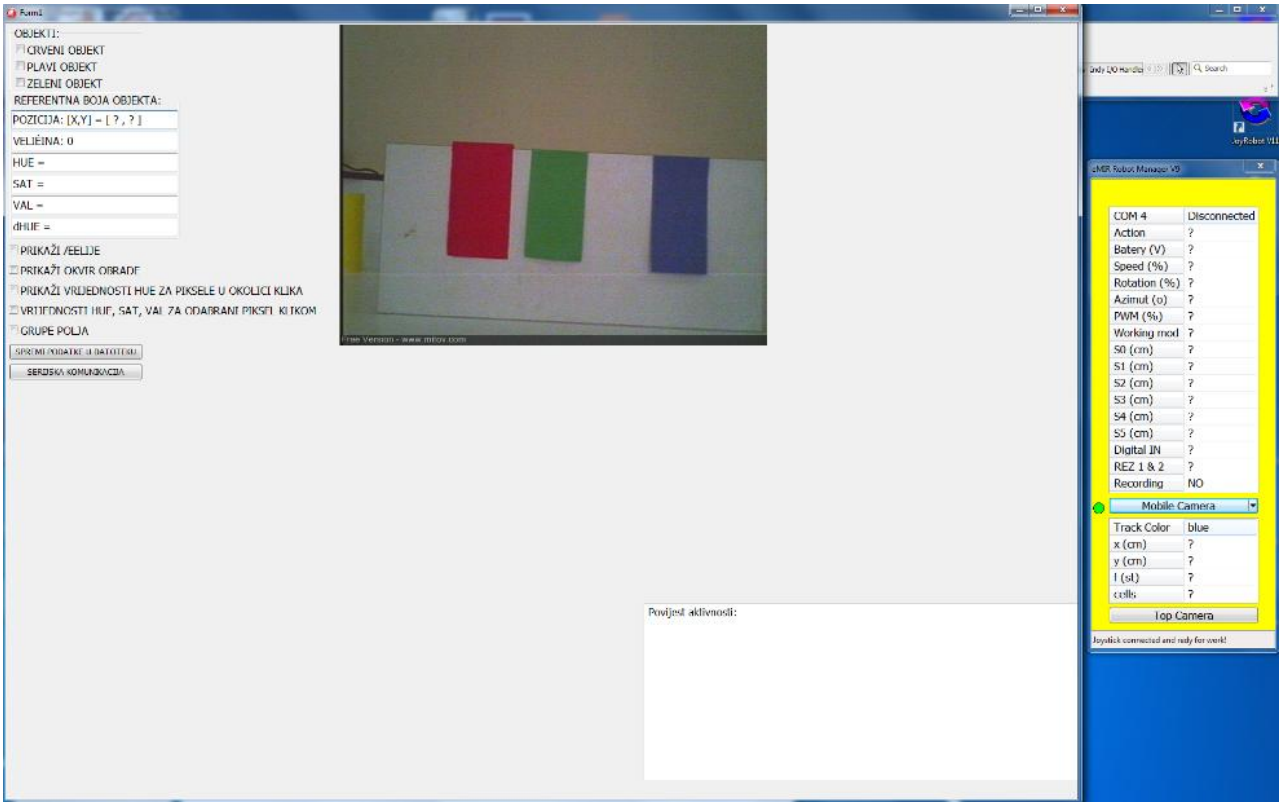
```
    ComPort1.WriteStr('*' + bbxxyy00nn00CS + '/');

     if Odabrana_Grupa = 0 then
       begin
         ComPort1.WriteStr('*?/');
         Exit;
       end;
   end
 else
   begin
     ComPort1.WriteStr('*X/');
     Exit;
   end;

end;
```

5.2. Rezultati s kamere robota

Slika 28. prikazuje položaj robota eMIR-a žute boje na poligonu, te postavljenih objekata za prepoznavanje po boji pred kamerom robota. Kamera se nalazi na žutoj ploči od robota eMIR-a.

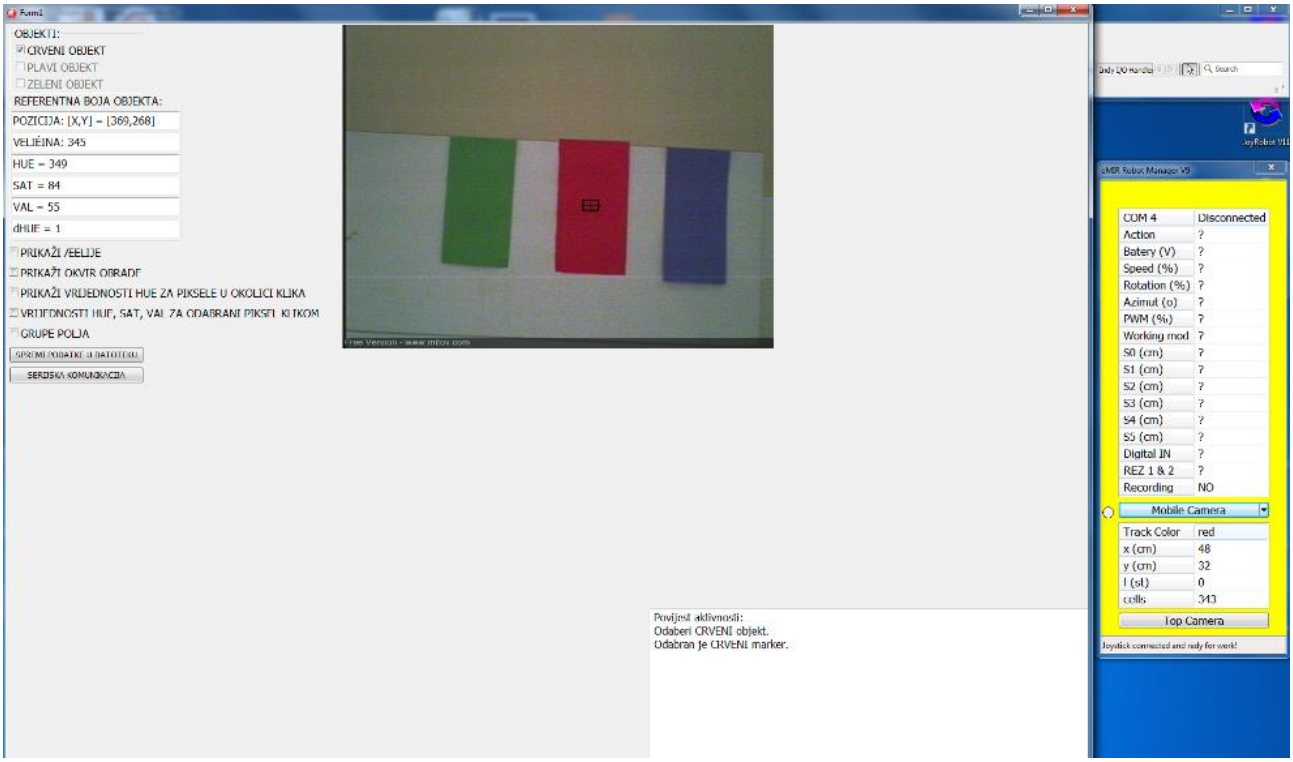


Slika 28. Mobilni robot sa web-kamerom



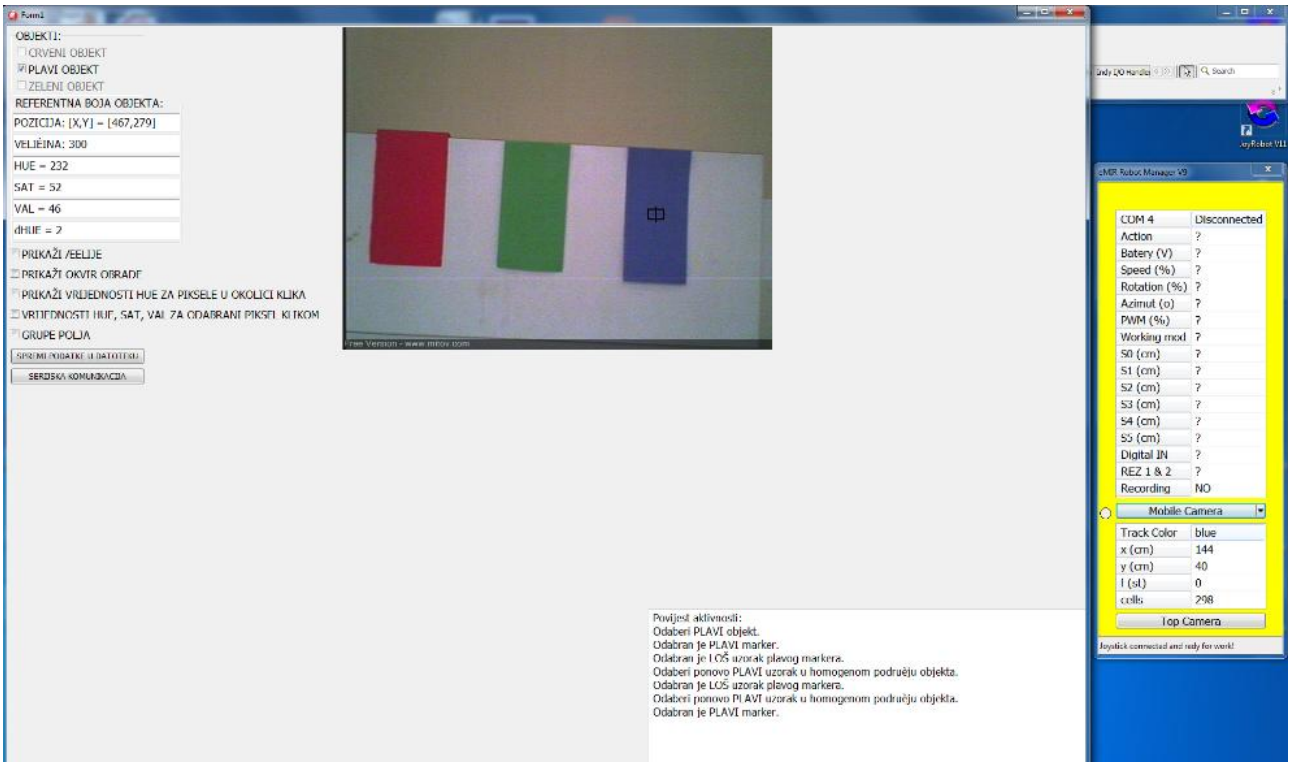
Slika 29. Sučelje u *Delphi*-u i komunikacijski prozor za robota eMIR-a (desno)

Slike 30., 31. i 32. pokazuju rezultate za objekte crvene, plave i zelene boje. Komunikacijski izbornik robota eMIR-a zahtjeva odabir tipa kamere (poligon ima jednu kameru poviše sebe na stropu koja daje informacije kao „satelit“, te kameru na robotu koja je mobilna). Odabiremo tipa kamere „Mobile camera“ u žutom prozoru, potrebno je također odabrati i koju boju objekta primamo podatke serijskom komunikacijom. Stoga će u rubrici *Track color* biti odabrana boja za koju će mobilni robot eMIR primati informacije položaja objekta i veličine.

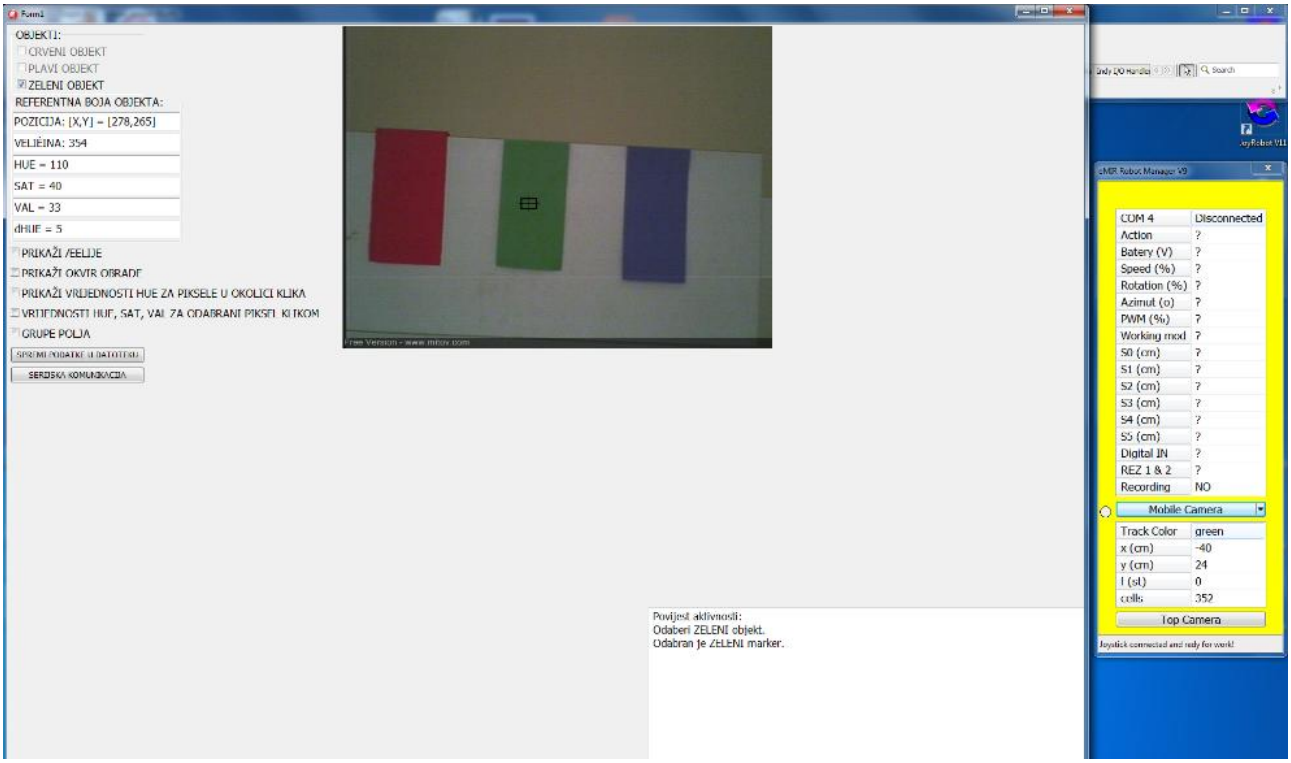


Slika 30. Kamera na robotu daje informacije za crveni objekt

Na slici 30. vidimo u *Delphi*-u podatke za crveni objekt, na kojem je pokazivač, da koordinate iznose „POZICIJA [X,Y] = [369, 268]“, te „VELIČINA: 345“. U desnom žutom prozoru vidimo podatke koje prima robot o položaju crvenog objekta, Xr = 48, Yr = 32, te *cells* = *343*. Ovdje vidimo razliku u vrijednostima koordinata, ne toliko veličine. U *Delphi*-u imamo ishodište kordinatnog sustava u gornjem lijevome kutu, dok kamera robota ima postavljeno ishodište u sredini kamere. Slika je dimenzije 640x480 piksela stoga centar slike je na X = 320, te Y = 240. Kada od vrijednosti u *Delphi*-u oduzmemo ovu razliku, X' = 369 – 320 = 49 dobijemo vrijednost koja je približno jednaka podatku koji je primio robot, Xr = 48. Također je i za koordinatu Y' = 268 – 240 = 28 što je vrijednost u broju piksela za 4 različita u odnosu na Yr = 32. Za veličinu objekta ono što je u *Delphi*-u pod „VELIČINA“ u žutom prozoru od robota je *cells*. Tu također vidimo da nema velike razlike. Potrebno je da koordinatni sustav robota bude postavljen u centar slike kamere kako bi robot znao na koju stranu se treba okrenuti prema objektu kako bi se točnije postavio naspram njega. Ove razlike u rezultatima su vidljive i kod primjera objekta plave i zelene boje na slici 31. i 32..



Slika 31. Kamera na robotu daje informacije za plavi objekt



Slika 32. Kamera na robotu daje informacije za zeleni objekt

6. ZAKLJUČAK

U ovom radu je iznesen primjer izrade programa kojim možemo pratiti odabrani objekt prema njegovoj boji (crvenoj, zelenoj, ili plavoj) pomoću sučelja koje je objektnim programiranjem napisano u programskom jeziku *Delphi*-u. O odabranom objektu na slici dobivenoj preko web-kamere, možemo dobiti informaciju o njegovoj boji, veličini te koordinate njegova težišta kao objekta. Uz par pokušaja iteracije nakon egzaktnog provjeravanja rezultata podešeni su parametri filtera za obradu slike. Tip filtera koji je odabran dovoljno je dobar za ovakvu vrstu problema te može poslužiti kod jednostavne navigacije mobilnih robota koji se navigiraju preko web-kamere.

Uz mentorstvo i sugestije ovim radom nije se uspjelo postići trajno prikazivanje rezultata. Također nije zadovoljen uvjet da se istovremeno prate tri objekta različite boje istodobno. Nastojanja da se riješi ovaj problem nisu donijeli rezultata s vremenom. Naime, tijekom perioda od prve minute kada se zorno prikazuju okvir pretraživanja ili prozor grupe odabranog objekta, te kada želimo prikazati ćelije koje sačinjavaju objekt izgubi se cijeli pokazivač i sve navedene opcije koje se u tom trenutku prikazuju. To je kod obrade slike za tri objekta različite boje uzrokovalo zamrzavanje slike i blokiranje aplikacije. Stoga ovdje je izneseno kako se relevantni podaci o boji jednog objekta, njegovoj veličini i koordinatama bez problema šalju serijskom komunikacijom. Gubljenje vizualne prezentacije onoga što se prati preko kamere ne pruža sigurnost u samo rješenje. Ovaj problem se nastojao ukloniti tako da sliku na kojoj se prikazuju rezultati se odvoji od one na kojoj se vrši obrada slike. Ta mala razlika u značajkama slike je nebitna ali nije doprinijela i dalje jasnijem vizualnom, tj. dugotrajnijem prikazu rezultata. Program je testiran i na drugom računalu s jačim svojstvima i u novijoj verziji *Delphi*-a, međutim pouzdanost prikazivanja vizualno rezultata nije poboljšana.

Ovo ponuđeno rješenje se može nadograđivati u smjeru da istovremeno pratimo tri objekta različite boje. Također može se nadograđivati problematikom prepoznavanja geometrijskih oblika kao što bi bili: krug, kvadar, pravokutnik, trokut.

LITERATURA

[1] Tomislav Horvat: Diplomski rad, Zagreb, 2013.

[2] Malik. J., online-tečaj: "Computer vision", 2012.

[3] http://lab405.fesb.hr/igraf/Frames/fP5_1.htm

[4] Program *Delphi*: Help

PRILOZI

I. CD-R disc

II. Popis procedura, funkcija i važnijih varijabli korištenih u *Delphi*-u

III. Tehnička dokumentacija

II. Popis procedura, funkcija i važnijih varijabli korištenih u *Delphi*-u

matricaRGB – matrica u kojoj su upisane vrijednosti RGB za svaki piksel slike s kamere

procedure slika_u_matricuRGB(slika_kamere: TBitmap); – procedura za upisivanje RGB vrijednosti slike u matricu *matricaRGB*

matricaHSV – matrica u kojoj su upisane vrijednosti HSV za svaki piksel prema matirci *matricaRGB*

procedure Definiranje_matriceHSV; – procedura za upisivanje HSV vrijednosti u matricu *matricaHSV*

function RGB2HSV (matricaRGB : TRGB) : THSV; – funkcija koja vrši transformaciju RGB vrijednosti u HSV vrijednosti

matricaHSV_polja7x7 – matrica koja sadrži uprosječene vrijednosti HSV za polja od 7x7 piksela

procedure Robusna_matricaHSV7x7; – procedura za upisivanje HSV vrijednosti u matricu *matricaHSV_polja7x7*

procedure obrada(slika_kamere: TBitmap); – procedura koja sadrži filter i u kojoj se vrši obrada slike

III.

Tehnička dokumentacija

UNIT - *glavni_prozor:*

```
unit glavni_prozor;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, CPort, VCL.LPControl,
  SLControlCollection, VLCommonDisplay, VLImageDisplay, Vcl.StdCtrls,
  Vcl.ExtCtrls, SLCommonFilter, TLBasicTimingFilter, TLClockGen,
  VLBasicGenericFilter, VLGenericFilter, VLDSCapture, Mitov.Types, LPComponent,
  VLCommonFilter, VLSnapshot, Math;

type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    GroupBox2: TGroupBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    VLSnapshot1: TVLSnapshot;
    VLDSCapture1: TVLDSCapture;
    VLGenericFilter1: TVLGenericFilter;
    TLClockGen1: TTLClockGen;
    UCITAJ_CIJELU_SLIKU: TTimer;
    UZIMANJE_SLIKE_ZA_OBRADU: TTimer;
    Memo1: TMemo;
    Memo2: TMemo;
    Button1: TButton;
    Button2: TButton;
    CheckBox4: TCheckBox;
    CheckBox5: TCheckBox;
    CheckBox6: TCheckBox;
    CheckBox7: TCheckBox;
    VLImageDisplay1: TVLImageDisplay;
    ComPort1: TComPort;
    CheckBox8: TCheckBox;
    Edit6: TEdit;
    PRIKAZ_GRUPA: TTimer;
    ISPIS_PARAMETARA: TTimer;
    KOMUNIKACIJA: TTimer;
    procedure UCITAJ_CIJELU_SLIKUTimer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure VLImageDisplay1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure CheckBox6Click(Sender: TObject);
    procedure CheckBox8Click(Sender: TObject);
    procedure PRIKAZ_GRUPATimer(Sender: TObject);
    procedure VLImageDisplay1MouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure CheckBox1Click(Sender: TObject);
    procedure CheckBox2Click(Sender: TObject);
    procedure CheckBox3Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure CheckBox4Click(Sender: TObject);
    procedure CheckBox5Click(Sender: TObject);
    procedure CheckBox7Click(Sender: TObject);
    procedure ISPIS_PARAMETARATimer(Sender: TObject);
    procedure VLGenericFilter1ProcessData(Sender: TObject;
      InBuffer: IVLImageBuffer; var OutBuffer: IVLImageBuffer;
      var SendOutputData: Boolean);
```

```
    procedure UZIMANJE_SLIKE_ZA_OBRADUTimer(Sender: TObject);
    procedure KOMUNIKACIJATimer(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

const
   centar_X = 320;
   centar_Y = 240;

type
  TKUT = packed record Kut, Sin_kut, Cos_kut: integer;  end;
  TKUT1 = array [0..360] of TKUT;
  PKUT = ^TKUT1;
  TRGB = packed record Blue, Green, Red: byte;  end;
  TRGB1 = array [0..32767] of TRGBQuad;
  PRGB = ^TRGB1;

var
  Form1: TForm1;
  slika_kamere, slika_za_obradu : TBitmap;
  PrikazGrupaPolja, imamo_sliku : Boolean;
  prikaz_celija_i_okviraPretrage : Boolean;
  prikazCelija, prikazOkvira, prikazHueVrijednosti, HueSatValVrijednostiPiksela : Boolean;
  tab_Sin_Cos : array [0..360] of TKUT;
  U_HUE, U_SAT, U_VAL, U_dHUE, U_aMmHUE : Integer;       // VRIJEDNOSTI SVAKOG UZORKA
  Ref_HUE, Ref_SAT, Ref_VAL, Ref_dHUE, Ref_aMmHUE, Ref_X, Ref_Y : Integer;  // VRIJEDNOSTI REFERENCE
  Tol_HUE, Tol_SAT, Tol_VAL, Tol_dHUE, Tol_aMmHUE : Integer;  // TOLERANCIJE ZA SVAKI UZORAK
  Boja_pretrage: TColor;
  Odabrana_boja : Integer;

implementation

uses obrada_slike, slika_u_matrici;

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var
  BINARNA, GRUPE : TextFile;
  x, y : integer;
  tekst : string;
begin
  AssignFile(BINARNA, 'BIN.txt'); Rewrite(BINARNA);
    for y := 0 to 67 do
      begin
        font.Size := 6;
        tekst := '';
        for x := 0 to 90 do
          if matricaHSV_polja7x7[y,x].Indikator_boje then
            tekst := tekst +  inttostr(1)
          else
            tekst := tekst +  inttostr(0);

        writeln(BINARNA, tekst);
      end;
  CloseFile(BINARNA);

  AssignFile(GRUPE, 'GRUPA.txt'); Rewrite(GRUPE);
  for y := 0 to 67 do
      begin
       font.Size := 6;
       tekst := '';
       for x := 0 to 90 do
          tekst := tekst + inttostr(matricaHSV_polja7x7[y,x].Broj_Grupe);
       writeln(GRUPE, tekst);
      end;
  CloseFile(GRUPE);

  slika_kamere.SaveToFile('SLIKA1.bmp');
```

```
  slika_za_obradu.SaveToFile('SLIKA2.bmp');
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  ComPort1.ShowSetupDialog;
  if ComPort1.Port <> '' then ComPort1.Open;
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
begin
  if CheckBox1.Checked = true then
    begin
      CheckBox2.Enabled := false;
      CheckBox3.Enabled := false;
      Odabrana_boja := 1;
      Memo1.Lines.Add('Odaberi CRVENI objekt.');
    end
   else
    begin
      CheckBox2.Enabled := True;
      CheckBox3.Enabled := True;
      Odabrana_boja := 0;
      Ref_SAT := 0;
      Ref_VAL := 0;
    end;
end;

procedure TForm1.CheckBox2Click(Sender: TObject);
begin
  if CheckBox2.Checked = true then
    begin
      CheckBox3.Enabled := false;
      CheckBox1.Enabled := false;
      Odabrana_boja := 2;
      Memo1.Lines.Add('Odaberi PLAVI objekt.');
    end
   else
    begin
      CheckBox3.Enabled := True;
      CheckBox1.Enabled := True;
      Odabrana_boja := 0;
      Ref_SAT := 0;
      Ref_VAL := 0;
    end;
end;

procedure TForm1.CheckBox3Click(Sender: TObject);
begin
    if CheckBox3.Checked = true then
    begin
      CheckBox1.Enabled := false;
      CheckBox2.Enabled := false;
      Odabrana_boja := 3;
      Memo1.Lines.Add('Odaberi ZELENI objekt.');
    end
   else
    begin
      CheckBox1.Enabled := True;
      CheckBox2.Enabled := True;
      Odabrana_boja := 0;
      Ref_SAT := 0;
      Ref_VAL := 0;
    end;
end;
```

```
procedure TForm1.CheckBox4Click(Sender: TObject);
begin
  if CheckBox4.Checked = true then
     prikazCelija := true
  else
     prikazCelija := false;
end;

procedure TForm1.CheckBox5Click(Sender: TObject);
begin
   if CheckBox5.Checked = true then
      prikazOkvira := true
   else
      prikazOkvira := false;
end;

procedure TForm1.CheckBox6Click(Sender: TObject);
begin
   if CheckBox6.Checked = true then
    begin
      CheckBox8.Enabled := false;
      prikazHueVrijednosti := true;
    end
   else
    begin
      CheckBox8.Enabled := true;
      prikazHueVrijednosti := false;
    end;
end;

procedure TForm1.CheckBox7Click(Sender: TObject);
begin
    if CheckBox7.Checked = true then
      HueSatValVrijednostiPiksela := true
   else
      HueSatValVrijednostiPiksela := false;
end;

procedure TForm1.CheckBox8Click(Sender: TObject);
begin
  if CheckBox8.Checked = true then
    begin
      CheckBox6.Enabled := false;
      PrikazGrupaPolja := true;
    end
   else
    begin
      CheckBox6.Enabled := true;
      PrikazGrupaPolja := false;
    end;
end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    slika_kamere.Free;
    slika_za_obradu.Free;
    ComPort1.Close;
end;

procedure TForm1.FormCreate(Sender: TObject);
var
 i : integer;
begin

    WindowState := wsMaximized;
    KeyPreview := True;
    prvi_prolaz := false;
    prikaz_celija_i_okviraPretrage := false;
    Odabrana_Grupa := 0;
    imamo_sliku := false;
```

```
    ComPort1.Port:= 'COM100';
    ComPort1.BaudRate:= br57600;
    ComPort1.Open;

    for i := 0 to 360 do
      begin
        tab_Sin_Cos[i].Kut := i;
        tab_Sin_Cos[i].Sin_kut := round(Sin(i*Pi/180) * 1000);
        tab_Sin_Cos[i].Cos_Kut := round(Cos(i*Pi/180) * 1000);
      end;

with VLImageDisplay1 do                    // pozicija i geometrija
begin
  Width := 640;
  Height := 480;
  Top := 3;
  Left := Screen.Width - VLImageDisplay1.Width - 5;
end;

with Memo1 do                              // pozicija i geometrija
begin
  Clear;
  Width := round(Screen.Width/2);
  Height := round(Screen.Height / 4.5);
  Top := Screen.Height - Memo1.Height - 65;
  Left := Screen.Width - 5 - Memo1.Width;
  Text := 'Povijest aktivnosti:' + sLineBreak;
  Font.Height := 20;
  ScrollBars := ssVertical;
end;

with Memo2 do                              // pozicija i geometrija
begin
  Clear;
  Width := 0;
  Height := 0;
  Top := Screen.Height;
  Left := 0;
end;

with GroupBox2 do
begin
  Width := 200;
  Height := 100;
  Top :=  5;
  Left := 5;
  Caption := 'OBJEKTI:';
  Font.Height := 20;
end;

with GroupBox1 do                          // pozicija i geometrija natpisa Orjentacije
begin
  Width := 255;
  Height := 220;
  Top :=  GroupBox2.Height + 5;                //screen.Height - Memo1.Height - 65 - GroupBox2.Height;
  Left := 5;
  Caption := 'REFERENTNA BOJA OBJEKTA:';
  Font.Height := 20;
  edit1.Top := 26;
  edit1.Height := 25;
  edit1.Width := 250;
  edit1.Font.Height := 20;
  Edit1.Text := 'POZICIJA: [X,Y] = ';
  edit2.Top := edit1.Top + edit1.Height + 7;
  edit2.Height := 25;
  edit2.Width := 250;
  edit2.Font.Height := 20;
  Edit2.Text := 'VELIČINA: ';
  edit3.Top := edit2.Top + edit2.Height + 7;
  edit3.Height := 25;
  edit3.Width := 250;
```

```
  edit3.Font.Height := 20;
  Edit3.Text := 'HUE = ';
  edit4.Top := edit3.Top + edit3.Height + 7;
  edit4.Height := 25;
  edit4.Width := 250;
  edit4.Font.Height := 20;
  Edit4.Text := 'SAT = ';
  edit5.Top := edit4.Top + edit4.Height + 7;
  edit5.Height := 25;
  edit5.Width := 250;
  edit5.Font.Height := 20;
  Edit5.Text := 'VAL = ';
  edit6.Top := edit5.Top + edit5.Height + 7;
  edit6.Height := 25;
  edit6.Width := 250;
  edit6.Font.Height := 20;
  Edit6.Text := 'dHUE = ';
end;

with CheckBox4 do                       // pozicija i geometrija "PRIKAZ ĆELIJE""
begin
  Width := 200;
  Height := 30;
  Top := GroupBox1.Height + GroupBox2.Height + 5;
  Left := 5;
  Font.Height := 20;
  Caption := 'PRIKAŽI ĆELIJE';
end;

with CheckBox5 do                       // pozicija i geometrija "PRIKAZ OKVIRA"
begin
  Width := 200;
  Height := 30;
  Top := CheckBox4.Top + CheckBox4.Height;
  Left := 5;
  Font.Height := 20;
  Caption := 'PRIKAŽI OKVIR OBRADE SLIKE';
end;

with CheckBox6 do                       // pozicija i geometrija "PRIKAŽI VRIJEDNOSTI HUE ZA PIKSELE U OKOLICI KLIKA"
begin
  Width := 500;
  Height := 30;
  Top := CheckBox5.Top + CheckBox5.Height;
  Left := 5;
  Font.Height := 20;
  Caption := 'PRIKAŽI VRIJEDNOSTI HUE ZA PIKSELE U OKOLICI KLIKA';
end;

with CheckBox7 do                       // pozicija i geometrija "'VRIJEDNOSTI HUE, SAT, VAL ZA ODABRANI PIKSEL KLIKOM"
begin
  Width := 500;
  Height := 30;
  Top := CheckBox6.Top + CheckBox6.Height;
  Left := 5;
  Font.Height := 20;
  Caption := 'VRIJEDNOSTI HUE, SAT, VAL ZA ODABRANI PIKSEL KLIKOM';
end;

with CheckBox8 do                       // pozicija i geometrija "GRUPE POLJA"
begin
  Width := 200;
  Height := 30;
  Top := CheckBox7.Top + CheckBox7.Height;
  Left := 5;
  Font.Height := 20;
  Caption := 'GRUPE POLJA';
end;
```

```
with CheckBox1 do                       // pozicija i geometrija "CRVENI OBJEKT"
begin
  Width := 190;
  Height := 30;
  Top := 20;
  Left := 10;
  Font.Height := 20;
  Caption := 'CRVENI OBJEKT';
end;

with CheckBox2 do                       // pozicija i geometrija "PLAVI OBJEKT"
begin
  Width := 190;
  Height := 30;
  Top := CheckBox1.Height + 15;
  Left := 10;
  Font.Height := 20;
  Caption := 'PLAVI OBJEKT';
end;

with CheckBox3 do                       // pozicija i geometrija "ZELENI OBJEKT"
begin
  Width := 190;
  Height := 30;
  Top := CheckBox1.Height*2 + 10;
  Left := 10;
  Font.Height := 20;
  Caption := 'ZELENI OBJEKT';
end;

with Button1 do                     // pozicija i geometrija gumba "SPREMI PODATKE U DATOTEKU"
begin
  Width := 200;
  Height := 25;
  Top := CheckBox8.Top + CheckBox8.Height + 5;
  Left := 5;
  Font.Height := 16;
  Caption := 'SPREMI PODATKE U DATOTEKU';
end;

with Button2 do                     // pozicija i geometrija gumba "SERIJSKA KOMUNIKACIJA"
begin
  Width := 200;
  Height := 25;
  Top := Button1.Top + Button1.Height + 5;
  Left := 5;
  Font.Height := 16;
  Caption := 'SERIJSKA KOMUNIKACIJA';
end;

  slika_kamere := TBitmap.Create;
  slika_kamere.PixelFormat := pf32bit;
  slika_kamere.Height := 480;
  slika_kamere.Width := 640;

  slika_za_obradu := TBitmap.Create;
  slika_za_obradu.PixelFormat := pf32bit;
  slika_za_obradu.Height := 480;
  slika_za_obradu.Width := 640;

end;

procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);
begin
  if key = char(27) then Halt;
end;
```

```
procedure TForm1.ISPIS_PARAMETARATimer(Sender: TObject);
begin
    if (Odabrana_boja = 0) or (Odabrana_Grupa = 0) then
      begin
        Edit1.Text := 'POZICIJA: [X,Y] = [ ? , ? ]';
        Edit2.Text := 'VELIČINA: 0';
      end
    else
      begin
        Edit1.Text := 'POZICIJA: [X,Y] = [' + Inttostr(X_koord_pokazivaca) + ',' + IntToStr(Y_koord_pokazivaca) + ']';
        Edit2.Text := 'VELIČINA: ' + inttostr(velicina);
      end;
end;

procedure TForm1.PRIKAZ_GRUPATimer(Sender: TObject);
var
 x11, y11 : integer;
 ll : string;
begin
if PrikazGrupaPolja = true then
begin
  with Memo2 do                         // pozicija i geometrija
    begin
      Clear;
      Width := 600;
      Height := 700;
      Top := 5;
      Left := Screen.Width - VLImageDisplay1.Width - Memo2.Width - 5;
      Font.Height := 8;
    end;

  Memo2.Clear;

  ll := '';
      for y11 := 0 to 67 do
      begin
        for x11 := 0 to 90 do
          ll := ll + inttostr(matricaHSV_polja7x7[y11,x11].Broj_Grupe) + ' ';
        memo2.Lines.Add(ll);
        ll := '';
      end;

  Memo1.Lines.Add('Odabrana grupa piksela ima redni broj: ' + IntToStr(Odabrana_Grupa) + '.');

end
else
  with Memo2 do                         // pozicija i geometrija
    begin
      Clear;
      Width := 0;
      Height := 0;
      Top := Screen.Height;
      Left := 0;
    end;
end;
```

```
procedure TForm1.KOMUNIKACIJATimer(Sender: TObject);
var
  bb, xx, yy, nn, CS : Byte;
  bbxxyy00nn00CS : string;
  razlika : Byte;
begin
  if Odabrana_boja <> 0 then
    begin
      if Odabrana_boja = 1 then bb := 1
        else if Odabrana_boja = 2 then bb := 6
          else bb := 4;

      if X_koord_pokazivaca >= centar_X then
        xx := round((X_koord_pokazivaca - centar_X) / 8)
      else
        xx := 255 + round((X_koord_pokazivaca - centar_X) / 8) + 1;

      if Y_koord_pokazivaca >= centar_Y then
        yy := round((Y_koord_pokazivaca - centar_Y) / 8)
      else
        yy := 255 + round((Y_koord_pokazivaca - centar_Y) / 8) + 1;

      if velicina <= 127 then
        nn := velicina
      else
        nn := round(velicina / 9) + 113;

      CS := Lo(256 - Lo(bb + xx + yy + nn));

     bbxxyy00nn00CS := IntToHex(bb,2) + IntToHex(xx,2) + IntToHex(yy,2) + '00' +  IntToHex(nn,2) + '00' + IntToHex(CS,2);

     ComPort1.WriteStr('*' + bbxxyy00nn00CS + '/');

      if Odabrana_Grupa = 0 then
        begin
          ComPort1.WriteStr('*?/');
          Exit;
        end;
    end
  else
    begin
      ComPort1.WriteStr('*X/');
      Exit;
    end;

end;

procedure TForm1.UCITAJ_CIJELU_SLIKUTimer(Sender: TObject);
begin
  prvi_prolaz := false;
end;

procedure TForm1.UZIMANJE_SLIKE_ZA_OBRADUTimer(Sender: TObject);
begin
  if imamo_sliku then
  begin
    obrada(slika_za_obradu);
    imamo_sliku := false;
  end;
end;
```

```
procedure TForm1.VLGenericFilter1ProcessData(Sender: TObject;
  InBuffer: IVLImageBuffer; var OutBuffer: IVLImageBuffer;
  var SendOutputData: Boolean);
var
  qq, ww, rac_q, rac_w : integer;
begin
    InBuffer.ToBitmap(slika_kamere);

    if imamo_sliku = false then
    begin
      InBuffer.ToBitmap(slika_za_obradu);
      slika_za_obradu.PixelFormat := pf32bit;
      imamo_sliku := true;
    end;

  if Odabrana_boja <> 0 then
  begin
    if Odabrana_Grupa <> 0 then
    begin
      slika_kamere.Canvas.Brush.Style := bsClear;
      slika_kamere.Canvas.Pen.Color := clBlack;
      slika_kamere.Canvas.Pen.Width := 2;
      slika_kamere.Canvas.Rectangle(X_koord_pokazivaca-12, Y_koord_pokazivaca-8, X_koord_pokazivaca+12, Y_koord_pokazivaca+8);
      slika_kamere.Canvas.Pen.Width := 1;
      slika_kamere.Canvas.MoveTo(X_koord_pokazivaca - 16, Y_koord_pokazivaca - 1);
      slika_kamere.Canvas.LineTo(X_koord_pokazivaca + 15, Y_koord_pokazivaca - 1);
      slika_kamere.Canvas.MoveTo(X_koord_pokazivaca - 1, Y_koord_pokazivaca - 12);
      slika_kamere.Canvas.LineTo(X_koord_pokazivaca - 1, Y_koord_pokazivaca + 11);
    end;

      if (prikazCelija = true) then
          begin
            slika_kamere.Canvas.Brush.Style := bsClear;
            slika_kamere.Canvas.Pen.Color := clBlack;
            slika_kamere.Canvas.Pen.Width := 1;
               for qq := 0 to 67 do
                  for ww := 0 to 90 do
                    if matricaHSV_polja7x7[qq,ww].Indikator_boje = true then
                    begin
                      rac_q :=  qq*7 + 2 - 4;
                      rac_w :=  ww*7 + 1 - 4;
                      slika_kamere.Canvas.Rectangle(rac_w - 4, rac_q - 4, rac_w + 4, rac_q + 4);
                    end;
          end;

        if (prikazOkvira = true) then
            begin
              slika_kamere.Canvas.Brush.Style := bsClear;
              slika_kamere.Canvas.Pen.Color := Boja_pretrage;
              slika_kamere.Canvas.Pen.Width := 2;
              slika_kamere.Canvas.Rectangle(prozor_provjere[1], prozor_provjere[3], prozor_provjere[2], prozor_provjere[4]);
              slika_kamere.Canvas.Pen.Width := 1;
              slika_kamere.Canvas.Rectangle(novi_XL_grupe, novi_YG_grupe, novi_XD_grupe, novi_YD_grupe);
            end;
  end;

    OutBuffer.FromBitmap(slika_kamere);
end;
```

```
procedure TForm1.VLImageDisplay1MouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
 x1, y1, x7, y7 : integer;
 l : string;
begin
if prikazHueVrijednosti then
begin
  with Memo2 do                          // pozicija i geometrija
    begin
      Clear;
      Width := 350;
      Height := 300;
      Top := 5;
      Left := Screen.Width - VLImageDisplay1.Width - memo2.Width - 5;
      Font.Height := 16;
    end;

  if X < 9 then X := 9;
  if Y < 8 then Y := 8;
  if X > 638 then X := 638;
  if Y > 478 then Y := 478;

  x7 := round(X/7);
  y7 := round(Y/7);

  Memo2.Clear;
  Memo2.Text := 'HUE PIKSELA NA MJESTU KILKA POKAZIVAČEM:' + sLineBreak;
  l := '';
      for y1 := 0 to 6 do
      begin
        for x1 := 0 to 6 do
            l := l + inttostr(round(matricaHSV[Y-3+y1,X-3+x1].Hue)) + ', ';
        memo2.Lines.Add(l);
        l := '';
      end;

  Memo2.Text := Memo2.Text + sLineBreak;
  Memo2.Text := Memo2.Text + 'HSV POLJA: 7x7 PIKSELA' + sLineBreak;
  l := '';
      for y1 := 0 to 6 do
      begin
        for x1 := 0 to 6 do
            l := l + inttostr(round(matricaHSV_polja7x7[y7-3+y1,x7-3+x1].Hue)) + ', ';
        memo2.Lines.Add(l);
        l := '';
      end;
end
else
  with Memo2 do                          // pozicija i geometrija
    begin
      Clear;
      Width := 0;
      Height := 0;
      Top := Screen.Height;
      Left := 0;
    end;
end;
```

```
procedure TForm1.VLImageDisplay1MouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  i, j : integer;
  x_i, y_j : integer;
  Hue, Sat, Val, d_HUE : Integer;
  R_Sin, R_Cos : integer;
  Uku_Tan : Real;
begin
      if X < 8 then X := 8;
      if Y < 9 then Y := 9;
      if X > 638 then X := 638;
      if Y > 478 then Y := 478;

      Hue := 0;
      Sat := 0;
      Val := 0;
      d_HUE := 0;

      X_koord_pokazivaca := X;
      Y_koord_pokazivaca := Y;

      if (matricaHSV[Y,X].Hue >= 340) or (matricaHSV[Y,X].Hue <= 40)  then
        begin
        R_Sin := 0;
        R_Cos := 0;
          for j := 0 to 6 do
          begin
            y_j := Y - 3 + j;
            for i := 0 to 6 do
              begin
                x_i := X - 3 + i;
                R_Sin := R_Sin + tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Sin_kut;
                R_Cos := R_Cos + tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Cos_kut;
              end;
          end;
          U_HUE := round(arctan(R_Sin/R_Cos)*180/PI);
          if U_HUE < 0 then U_HUE := U_HUE + 360;

          Uku_Tan := 0;
          for j := 0 to 6 do
          begin
            y_j := Y - 3 + j;
            for i := 0 to 6 do
              begin
                x_i := X - 3 + i;
                Uku_Tan := Uku_Tan + power((tan(U_HUE/180*PI) –
                                    tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Sin_kut/tab_Sin_Cos[matricaHSV[y_j,x_i].Hue].Cos_kut),2);
              end;
          end;
          U_dHUE := round(arctan(power(Uku_Tan/49,0.5))*180/PI);
        end
      else
        begin
          for j := 0 to 6 do
          begin
            y_j := Y - 3 + j;
            for i := 0 to 6 do
              begin
                x_i := X - 3 + i;
                Hue := Hue + matricaHSV[y_j,x_i].Hue;
              end;
          end;
          U_HUE := round(Hue / 49);
```

```
      for j := 0 to 6 do
      begin
        y_j := Y - 3 + j;
        for i := 0 to 6 do
          begin
            x_i := X - 3 + i;
            d_HUE := d_HUE + round(abs(U_HUE - matricaHSV[y_j,x_i].Hue));
          end;
      end;
      U_dHUE := round(d_HUE / 49);
    end;

  for j := 0 to 6 do
    begin
      y_j := Y - 3 + j;
      for i := 0 to 6 do
        begin
          x_i := X - 3 + i;
          Sat := Sat + matricaHSV[y_j,x_i].Sat;
          Val := Val + matricaHSV[y_j,x_i].Val;
        end;
    end;

    U_SAT := round(Sat / 49);
    U_VAL := round(Val / 49);

    Ref_Hue := U_HUE;
    Ref_Sat := U_SAT;
    Ref_Val := U_VAL;
    Ref_dHue := U_dHUE;

    if HueSatValVrijednostiPiksela then
     Memo1.Lines.Add('HUE = ' + IntToStr(matricaHSV[Y,X].Hue) + ', ' + 'SAT = ' + IntToStr(matricaHSV[Y,X].Sat) + ', ' +
                                'VAL = ' + IntToStr(matricaHSV[Y,X].Val) + '.');

    if Odabrana_boja <> 0 then
    begin
        if Odabrana_boja = 1 then
        begin
         if (Ref_Hue >= 340) or (Ref_Hue <= 40) then
           begin
             if (Ref_Val >= 40) and (Ref_Sat >= 40) and (Ref_dHue <= 8) then
               begin
                 Tol_HUE := 8;
                 Tol_SAT := 15;
                 Tol_VAL := 20;
                 Tol_dHUE := 5;
                 Tol_aMmHUE := 15;
                 Edit3.Text := 'HUE = ' + inttostr(Ref_Hue);
                 Edit4.Text := 'SAT = ' + inttostr(Ref_Sat);
                 Edit5.Text := 'VAL = ' + inttostr(Ref_Val);
                 Edit6.Text := 'dHUE = ' + inttostr(Ref_dHue);
                 Boja_pretrage := clRed;
                 Memo1.Lines.Add('Odabran je CRVENI marker.');
               end
             else
               begin
                 Memo1.Lines.Add('Odabran je LOŠ uzorak crvenog markera.');
                 Memo1.Lines.Add('Odaberi ponovo CRVENI uzorak u homogenom području objekta.');
               end;
           end
           else
             begin
               Memo1.Lines.Add('ODABRANI UZORAK NIJE TRAŽENI CRVENI OBJEKT!');
               Memo1.Lines.Add('Odaberite ponovo marker.');
             end;
        end
```

```
          else IF (Odabrana_boja = 2) then
          if (Ref_Hue >= 190) and (Ref_Hue <= 250) then
            if (Ref_Val >= 40) and (Ref_Sat >= 40) and (Ref_dHue <= 8) then
              begin
                Tol_HUE := 5;
                Tol_SAT := 10;
                Tol_VAL := 20;
                Tol_dHUE := 5;
                Tol_aMmHUE := 15;
                Edit3.Text := 'HUE = ' + inttostr(Ref_Hue);
                Edit4.Text := 'SAT = ' + inttostr(Ref_Sat);
                Edit5.Text := 'VAL = ' + inttostr(Ref_Val);
                Edit6.Text := 'dHUE = ' + inttostr(Ref_dHue);
                Boja_pretrage := clBlue;
                Memo1.Lines.Add('Odabran je PLAVI marker.');
              end
            else
              begin
                Memo1.Lines.Add('Odabran je LOŠ uzorak plavog markera.');
                Memo1.Lines.Add('Odaberi ponovo PLAVI uzorak u homogenom području objekta.');
              end
          else
            begin
              Memo1.Lines.Add('ODABRANI UZORAK NIJE TRAŽENI PLAVI OBJEKT!');
              Memo1.Lines.Add('Odaberite ponovo marker.');
            end

        else if (Odabrana_boja = 3) then
        if(Ref_Hue >= 90) and (Ref_Hue <= 150) then
          begin
            if (Ref_Val >= 30) and (Ref_Sat >= 20) and (Ref_dHue <= 8) then
              begin
                Tol_HUE := 15;
                Tol_SAT := 15;
                Tol_VAL := 20;
                Tol_dHUE := 5;
                Tol_aMmHUE := 15;
                Edit3.Text := 'HUE = ' + inttostr(Ref_Hue);
                Edit4.Text := 'SAT = ' + inttostr(Ref_Sat);
                Edit5.Text := 'VAL = ' + inttostr(Ref_Val);
                Edit6.Text := 'dHUE = ' + inttostr(Ref_dHue);
                Boja_pretrage := clGreen;
                Memo1.Lines.Add('Odabran je ZELENI marker.');
              end
            else
              begin
                Memo1.Lines.Add('Odabran je LOŠ uzorak zelenog markera.');
                Memo1.Lines.Add('Odaberi ponovo ZELENI uzorak u homogenom području objekta.');
              end;
          end
        else
          begin
            Memo1.Lines.Add('ODABRANI UZORAK NIJE TRAŽENI ZELENI OBJEKT!');
            Memo1.Lines.Add('Odaberite ponovo marker.');
          end;
    end
      else
          begin
            Memo1.Lines.Add('ODABRANI UZORAK NIJE JEDAN OD TRAŽENIH MARKERA!');
            Memo1.Lines.Add('Odaberite ponovo marker.');
          end;

  prvi_prolaz := false;
end;

end.
```

UNIT – *obrada_slika*:

```
unit obrada_slike;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, VLImageDisplay, VCL.LPControl, System.UITypes,
  SLControlCollection, VLCommonDisplay, VLDSImageDisplay, VLCommonLogger,
  VLDSVideoLogger, VLBasicGenericFilter, VLGenericFilter, SLCommonFilter,
  TLBasicTimingFilter, TLClockGen, LPComponent, VLCommonFilter, VLSnapshot,
  Mitov.Types, VLDSCapture, Vcl.StdCtrls, Vcl.Menus, Vcl.ExtCtrls, VLCanny, Math;

procedure obrada(slika_kamere: TBitmap);

type
  TOkvir_pretrage = packed record XL, XD, YG, YD : integer;  end;
  TOkvir_pretrage1 = array [0..1] of TOkvir_pretrage;
  POkvir_pretrage = ^TOkvir_pretrage1;

var
    X_koord_pokazivaca, Y_koord_pokazivaca, velicina : integer;
    prozor_provjere : array [1..4] of integer;
    XL_provjere, XD_provjere, YG_provjere, YD_provjere : integer;
    novi_XL_grupe, novi_XD_grupe, novi_YG_grupe, novi_YD_grupe : integer;
    razmak_pretrage, razmak_grupe, razmak_razlike : Integer;
    prvi_prolaz : Boolean;
    matrica_pretrage : TOkvir_pretrage;
    brojac, Odabrana_Grupa : integer;
    novi_XL_pretrage, novi_XD_pretrage, novi_YG_pretrage, novi_YD_pretrage : integer;

implementation

uses glavni_prozor, slika_u_matrici;

procedure obrada(slika_kamere: TBitmap);
var
  q, w, i : integer;
  rac_q, rac_w : integer;
begin

    razmak_grupe := 21;
    razmak_pretrage := 35;
    razmak_razlike := razmak_pretrage - razmak_grupe;

    if prvi_prolaz = true then
      begin
        XL_provjere := round((prozor_provjere[1] + 1)/7);
        XD_provjere := round((prozor_provjere[2] + 1)/7);
        YG_provjere := round((prozor_provjere[3] + 2)/7);
        YD_provjere := round((prozor_provjere[4] + 2)/7);
        matrica_pretrage.XL := prozor_provjere[1];
        matrica_pretrage.XD := prozor_provjere[2];
        matrica_pretrage.YG := prozor_provjere[3];
        matrica_pretrage.YD := prozor_provjere[4];
      end
    else
      begin
        XL_provjere := 0;
        XD_provjere := 90;
        YG_provjere := 0;
        YD_provjere := 67;
        matrica_pretrage.XL := 1;
        matrica_pretrage.XD := 638;
        matrica_pretrage.YG := 2;
        matrica_pretrage.YD := 478;
      end;
```

```
 slika_u_matricuRGB(slika_kamere);

if (Odabrana_boja <> 0) then
  begin
    novi_XL_pretrage := 638;
    novi_XD_pretrage := 1;
    novi_YG_pretrage := 478;
    novi_YD_pretrage := 2;

    novi_XL_grupe := 638;
    novi_XD_grupe := 1;
    novi_YG_grupe := 478;
    novi_YD_grupe := 2;

    i := 0;

    for q := YG_provjere to YD_provjere do
    begin
      rac_q := q*7 + 2 - 3;
      for w := XL_provjere to XD_provjere do
      begin
        rac_w := w*7 + 1 - 3;
        if (abs(Ref_Val - matricaHSV_polja7x7[q,w].Val) <= Tol_VAL) and
           (abs(Ref_Sat - matricaHSV_polja7x7[q,w].Sat) <= Tol_SAT) and
           (abs(Ref_dHUE - matricaHSV_polja7x7[q,w].dHUE) <= Tol_dHUE) and
           (abs(Ref_HUE - matricaHSV_polja7x7[q,w].Hue) <= Tol_HUE) then
            begin
              matricaHSV_polja7x7[q,w].Indikator_boje := true;
              if novi_XL_pretrage > rac_w then novi_XL_pretrage := rac_w;
              if novi_XD_pretrage < rac_w then novi_XD_pretrage := rac_w;
              if novi_YG_pretrage > rac_q then novi_YG_pretrage := rac_q;
              if novi_YD_pretrage < rac_q then novi_YD_pretrage := rac_q;
            end
          else
              matricaHSV_polja7x7[q,w].Indikator_boje := false;
      end;
    end;

    for q := YG_provjere to YD_provjere do     //Grupiranje polja koja odgovaraju REF vrijednosti
      for w := XL_provjere to XD_provjere do
          if matricaHSV_polja7x7[q,w].Indikator_boje then
              begin
                if (matricaHSV_polja7x7[q-1,w-1].Indikator_boje = false) and
                   (matricaHSV_polja7x7[q-1,w].Indikator_boje = false) and
                   (matricaHSV_polja7x7[q-1,w+1].Indikator_boje = false) and
                   (matricaHSV_polja7x7[q,w-1].Indikator_boje = false) then
                    begin
                      i := i + 1;
                      matricaHSV_polja7x7[q,w].Broj_grupe := i;
                    end
                else
                    if matricaHSV_polja7x7[q-1,w-1].Indikator_boje then
                        matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w-1].Broj_grupe
                    else if matricaHSV_polja7x7[q-1,w].Indikator_boje then
                        matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w].Broj_grupe
                    else if matricaHSV_polja7x7[q-1,w+1].Indikator_boje then
                        matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q-1,w+1].Broj_grupe
                    else if matricaHSV_polja7x7[q,w-1].Indikator_boje then
                        matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w-1].Broj_grupe;
              end
          else
            matricaHSV_polja7x7[q,w].Broj_grupe := 0;
```

```
 for q := YD_provjere downto YG_provjere do    //Grupiranje polja koja odgovaraju REF vrijednosti
     for w := XD_provjere downto XL_provjere do
       if matricaHSV_polja7x7[q,w].Broj_grupe <> 0 then
         begin
                 if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w+1].Broj_grupe) and
(matricaHSV_polja7x7[q+1,w+1].Broj_grupe <> 0) then
                     matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w+1].Broj_grupe
                 else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w].Broj_grupe) and
(matricaHSV_polja7x7[q+1,w].Broj_grupe <> 0) then
                     matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w].Broj_grupe
                 else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q+1,w-1].Broj_grupe) and
(matricaHSV_polja7x7[q+1,w-1].Broj_grupe <> 0) then
                     matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q+1,w-1].Broj_grupe
                 else if (matricaHSV_polja7x7[q,w].Broj_grupe <> matricaHSV_polja7x7[q,w+1].Broj_grupe) and
(matricaHSV_polja7x7[q,w+1].Broj_grupe <> 0) then
                     matricaHSV_polja7x7[q,w].Broj_grupe := matricaHSV_polja7x7[q,w+1].Broj_grupe
         end;

  Odabrana_Grupa :=  matricaHSV_polja7x7[round(Y_koord_pokazivaca / 7),round(X_koord_pokazivaca / 7)].Broj_grupe;

  for q := YG_provjere to YD_provjere do
   begin
    rac_q := q*7 + 2 - 3;
    for w := XL_provjere to XD_provjere do
     begin
      rac_w := w*7 + 1 - 3;
      if matricaHSV_polja7x7[q,w].Broj_grupe = Odabrana_Grupa then
         begin
           if novi_XL_grupe > rac_w then novi_XL_grupe := rac_w;
           if novi_XD_grupe < rac_w then novi_XD_grupe := rac_w;
           if novi_YG_grupe > rac_q then novi_YG_grupe := rac_q;
           if novi_YD_grupe < rac_q then novi_YD_grupe := rac_q;
         end;
     end;
   end;

  X_koord_pokazivaca := round((novi_XL_grupe + novi_XD_grupe)/2);
  Y_koord_pokazivaca := round((novi_YG_grupe + novi_YD_grupe)/2);

    if (novi_XL_pretrage - razmak_pretrage) < 1 then prozor_provjere[1] := 1
      else  prozor_provjere[1] := novi_XL_pretrage - razmak_pretrage;
    if (novi_XD_pretrage + razmak_pretrage) > 638 then prozor_provjere[2] := 638
      else  prozor_provjere[2] := novi_XD_pretrage + razmak_pretrage;
    if (novi_YG_pretrage - razmak_pretrage) < 2 then prozor_provjere[3] := 2
      else  prozor_provjere[3] := novi_YG_pretrage - razmak_pretrage;
    if (novi_YD_pretrage + razmak_pretrage) > 478 then prozor_provjere[4] := 478
      else  prozor_provjere[4] := novi_YD_pretrage + razmak_pretrage;

    if (novi_XL_grupe - razmak_grupe) < 1 then novi_XL_grupe := 1
      else  novi_XL_grupe := novi_XL_grupe - razmak_grupe;
    if (novi_XD_grupe + razmak_grupe) > 638 then novi_XD_grupe := 638
      else  novi_XD_grupe := novi_XD_grupe + razmak_grupe;
    if (novi_YG_grupe - razmak_grupe) < 2 then novi_YG_grupe := 2
      else  novi_YG_grupe := novi_YG_grupe - razmak_grupe;
    if (novi_YD_grupe + razmak_grupe) > 478 then novi_YD_grupe := 478
      else  novi_YD_grupe := novi_YD_grupe + razmak_grupe;

   if Odabrana_Grupa = 0 then
   begin

    prozor_provjere[1] := 1 + razmak_razlike;
    prozor_provjere[2] := 638 - razmak_razlike;
    prozor_provjere[3] := 2 + razmak_razlike;
    prozor_provjere[4] := 478 - razmak_razlike;

    novi_XL_grupe := 1 + 2*razmak_razlike;
    novi_XD_grupe := 638 - 2*razmak_razlike;
    novi_YG_grupe := 2 + 2*razmak_razlike;
    novi_YD_grupe := 478 - 2*razmak_razlike;

   end;
```

```
    for q := 0 to 67 do
      for w := 0 to 90 do
          if matricaHSV_polja7x7[q,w].Broj_grupe = Odabrana_Grupa then  brojac := brojac + 1;

  velicina := brojac;

  if (velicina < 9) then
      prvi_prolaz := false
  else
      prvi_prolaz := true;

   brojac := 0;
end;

end;

end.
```

UNIT – *slika_u_matrici:*

```
unit slika_u_matrici;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, VLImageDisplay, VCL.LPControl, System.UITypes,
  SLControlCollection, VLCommonDisplay, VLDSImageDisplay, VLCommonLogger,
  VLDSVideoLogger, VLBasicGenericFilter, VLGenericFilter, SLCommonFilter,
  TLBasicTimingFilter, TLClockGen, LPComponent, VLCommonFilter, VLSnapshot,
  Mitov.Types, VLDSCapture, Vcl.StdCtrls, Vcl.Menus, Vcl.ExtCtrls, VLCanny, Math;

procedure slika_u_matricuRGB(slika_kamere: TBitmap);

type
  TRGB = packed record Blue, Green, Red: byte;  end;
  THSV = packed record Hue , Sat , Val : Integer; end;
  THSV7 = packed record Hue , Sat , Val, dHUE, aMmHUE, Broj_Grupe : Integer; Indikator_boje: boolean; end;
  TRGB1 = array [0..32767] of TRGBQuad;
  PRGB = ^TRGB1;
  THSV1 = array [0..32767] of THSV;
  PHSV = ^THSV1;
  THSV51 = array [0..32767] of THSV7;
  PHSV5 = ^THSV7;

var
  matricaRGB : array [0..479, 0..639] of TRGB;
  matricaHSV : array [0..479, 0..639] of THSV;
  matricaHSV_polja7x7 : array [0..67, 0..90] of THSV7;
  grubi_Hue, grubi_Sat, grubi_Val : Integer;
  slika_kamere: TBitmap;

implementation

uses glavni_prozor, obrada_slike;

//---------------------------------------------------------------------------
//             FUNKCIJA PRETVORBE PIKSELA IZ RGB U HSV VRIJEDNOST
//---------------------------------------------------------------------------
function RGB2HSV (matricaRGB : TRGB) : THSV;
var
  MinRGB, MaxRGB, Delta : Double;
  H , S , V : Double ;
begin
  H := 0.0 ;
  MinRGB := Min (Min( matricaRGB.Red, matricaRGB.Green ), matricaRGB.Blue);
  MaxRGB := Max (Max( matricaRGB.Red, matricaRGB.Green ), matricaRGB.Blue);
  Delta := ( MaxRGB - MinRGB );
  V := MaxRGB ;

  If V = 0.0 then
    S := 0.0
  else
    S := Delta / V ;

  If (S <> 0.0) then
    begin
      If matricaRGB.Red = V then
        H := 60.0 * (matricaRGB.Green - matricaRGB.Blue) / Delta
      else If matricaRGB.Green = V then
        H := 120.0 + 60.0 * (matricaRGB.Blue - matricaRGB.Red) / Delta
      else
        H := 240.0 + 60.0 * (matricaRGB.Red - matricaRGB.Green) / Delta
    End
  else
  H := 0;

 If H < 0.0 then    H := H + 360.0;
```

```
  If H = 360 then H := 0;

  with Result Do
    begin
      Hue := round(H) ;          // Hue -> 0..360
      Sat := round(S * 100);     // Saturation -> 0..100 %
      Val := round(V / 2.55);    // Value - > 0..100 %
    end;
end;

//------------------------------------------------------------------------
// ROBUSNA MATRICA HSV POLJA 7x7
//------------------------------------------------------------------------
procedure Robusna_matricaHSV7x7;
var
  x, y, xa, ya : integer;
  xx, yy, xi, yj : integer;
  kk, ll, dH : integer;
begin
  yy := matrica_pretrage.YG + 7;
  while yy <= matrica_pretrage.YD do
   begin
   xx := matrica_pretrage.XL + 7;
   while xx <= matrica_pretrage.XD do
      begin
      grubi_hue := 0;
      grubi_sat := 0;
      grubi_val := 0;
      dH := 0;

      for y := 0 to 6 do
        begin
          yj := yy - 1 - y;
          for x := 0 to 6 do
             begin
                 xi := xx - 1 - x;
                 grubi_hue := grubi_hue + matricaHSV[yj,xi].Hue;
                 grubi_sat := grubi_sat + matricaHSV[yj,xi].Sat;
                 grubi_val := grubi_val + matricaHSV[yj,xi].Val;
             end;
        end;

      yj := round((yy)/7-1);
      xi := round((xx)/7-1);
      matricaHSV_polja7x7[yj,xi].Hue := round(grubi_hue/49);
      matricaHSV_polja7x7[yj,xi].Sat := round(grubi_sat/49);
      matricaHSV_polja7x7[yj,xi].Val := round(grubi_val/49);

      for ya := 0 to 6 do
        begin
          kk := yy - 1 - ya;
          for xa := 0 to 6 do
            begin
              ll := xx - 1 - xa;
              dH := dH + abs(matricaHSV_polja7x7[yj,xi].Hue - matricaHSV[kk,ll].Hue);
            end;
        end;

      matricaHSV_polja7x7[yj,xi].dHUE := round(dH/49);

      inc(xx,7);
      end;
   inc(yy,7);
   end;
end;
```

```
//-------------------------------------------------------------------------------
// MATRICA HSV
//-------------------------------------------------------------------------------
procedure Definiranje_matriceHSV;
var
  xx,yy: integer;
begin
   for yy := matrica_pretrage.YG to matrica_pretrage.YD do
     for xx := matrica_pretrage.XL to matrica_pretrage.XD do
        begin
          matricaHSV[yy,xx] := RGB2HSV(matricaRGB[yy,xx]);
        end;
   Robusna_matricaHSV7x7;
end;         //matrica sadrži HSV vrijednosti

//-------------------------------------------------------------------------------
// MATRICA RGB
//-------------------------------------------------------------------------------
procedure slika_u_matricuRGB(slika_kamere: TBitmap);
var
  x,y: integer;
  linija: PRGB;
begin
   for y := matrica_pretrage.YG to matrica_pretrage.YD do
     begin
       linija := slika_kamere.ScanLine[y];
       for x := matrica_pretrage.XL to matrica_pretrage.XD do
         begin
           matricaRGB[y,x].Red := linija[x].rgbRed;
           matricaRGB[y,x].Green := linija[x].rgbGreen;
           matricaRGB[y,x].Blue := linija[x].rgbBlue;
         end;
      end;
   Definiranje_matriceHSV;
end;                                  //matrica sadrži RGB vrijednosti

end.
```