Detekcija objekta uz pomoć Web kamere i OpenCV-a

Pačarek, Marko

Undergraduate thesis / Završni rad

2017

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

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:325936

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-25**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Preddiplomski sveučilišni studij računarstva

DETEKCIJA OBJEKTA UZ POMOĆ WEB KAMERE I OPENCV-A

Završni rad

Marko Pačarek

Osijek, 2017

SADRŽAJ:

1. UVOD 1
1.1 Zadatak završnog rada 2
2. TEHNOLOGIJE I ALATI 3
2.1 C# 3
2.2 Microsoft Visual Studio 5
2.3 OpenCV / EmguCV 6
3. APLIKACIJA 8
3.1 Izrada aplikacije 8
3.2 Rezultati aplikacije 13
3.3 Problemi i izazovi 17
4. ZAKLJUČAK 18
LITERATURA 19
SAŽETAK 20
ABSTRACT 21
ŽIVOTOPIS 22

1. UVOD

Zadatak ovog završnog rada je izraditi *desktop* aplikaciju koja će omogućiti detekciju objekta odabrane boje u stvarnom vremenu pomoću Web kamere. Za oblik objekta koji se mogu detektirati izabran je oblik kružnice, a za boje koje se mogu odabrati izabrane su standardne boje koje se mogu pronaći u prirodi (dugine boje). Radi lakšeg snalaženja i boljeg izgleda aplikacije korištene su dodatne funkcije za zrcaljenje slike i detekciju rubova na slici.

Prilikom izrade završnog rada korištena su znanja stečena na fakultetu (iz kolegija „Programiranje 1“ , „Programiranje 2“ , „Objektno orijentirano programiranje“ i „Algoritmi i strukture podataka“) , kao i znanja stečena samostalnim radom iz područja računalnog vida (OpenCV/ EmguCV). U konzultaciji sa mentorom dogovoreno je da se aplikacija napravi u C# programskom jeziku radi jednostavnije izrade grafičkog sučelja. Za pisanje, prevođenje i testiranje koda korišteno je razvojno okruženje Microsoft Visual Studio, na kojemu je rađeno većina laboratorijskih vježbi na kolegijima programiranja. Za pozivanje funkcija koje obuhvaćaju područja računalnog vida korišten je skup biblioteka „EmguCV“. EmguCV je posebna inačica OpenCV-a koju koristimo za pozivanje funkcija računalnog vida u C#-u.

Završni rad je koncipiran tako da će se u početku glavnog dijela rada opisati tehnologije i alati, kao i teorijska podloga za izradu aplikacije. Nakon toga će biti opisana i sama aplikacija, a to uključuje: postupak izrade i rezultate pokretanja aplikacije. Glavni dio biti će zaključen ukazivanjem na probleme prilikom izrade i metode kojima su neki od njih riješeni. U zaključku rada daje se osvrt na ciljeve postavljene u zadatku završnog rada i postignute rezultate. Također su navedena i područja moguće primjene kao i prijedlozi za daljnje unaprjeđenje aplikacije u budućnosti.

1.1 Zadatak završnog rada

U okviru ovog završnog rada potrebno je upoznati se s metodama i algoritmima za detekciju objekta u stvarnom vremenu pomoću Web kamere. Opisati alate i tehnologije potrebne za izradu desktop aplikacije te se upoznati s bibliotekom OpenCV i njenim mogućnostima koja će vam pomoći u implementaciji rješenja. Potrebno je izraditi *desktop* aplikaciju koja će omogućiti detekciju objekta odabranog RGB spektra u stvarnom vremenu. Rješenje implementirati u C++/ C# programskom jeziku uz pomoć OpenCV/ EmguCV biblioteke.

2. TEHNOLOGIJE I ALATI

Kao što je navedeno u uvodu, za izradu aplikacije potrebna su znanja iz:

- C# - programskog jezika koji se pokazao kao najpraktičniji jezik za izradu aplikacije ovoga tipa zbog jednostavne izrade grafičkog sučelja aplikacije
- Microsoft Visual Studio-a – razvojnog okruženja koje služi za pisanje, prevođenje i testiranje koda
- OpenCV / EmguCV-a – skup biblioteka za pozivanje funkcija iz područja računalnog vida

2.1 C#

Programski jezik C# (*CSharp*) osmišljen je da bi bio jednostavan, siguran, moderan, objektno orijentirani jezik visokih performansi za .NET platformu. Jedan je od mlađih programskih jezika. Razvijen je od strane Andersa Hejlsberga, Scotta Wiltamutha, Petera Goldea i njihovog tima u tvrtki Microsoft, a predstavljen je 2002. godine kao sastavni dio MS.NET Framework 1.0. Prvo ime za C# bilo je „Cool“ (*C-like Object Oriented Language*) koje je promijenjeno godinu dana poslije. Organizacija i funkcionalnost ovog jezika temeljene su najviše na C-u (visoke performanse), C++-u (objektno orijentirana struktura) te na programskom jeziku Java (visoka razina sigurnosti). [1]

Svaki podatak u C# programskom jeziku zapravo je enkapsuliran u objekt neke klase, stoga možemo reći da je C# u potpunosti objektno orijentiran. C# je preuzeo sve dobre karakteristike C++ i Jave, što ga čini dobrim „potpuno objektno orijentiranim jezikom“ koji omogućava izradu vizualnih aplikacija korisnicima koji i nemaju prevelika znanja i iskustva u programiranju. Jednostavno je prikazan i način na koji nastaju objektne i vizualne aplikacije, i vrlo je lako objasniti korisnicima kako kreirati takve aplikacije te ih upravljati i koristiti. [2]

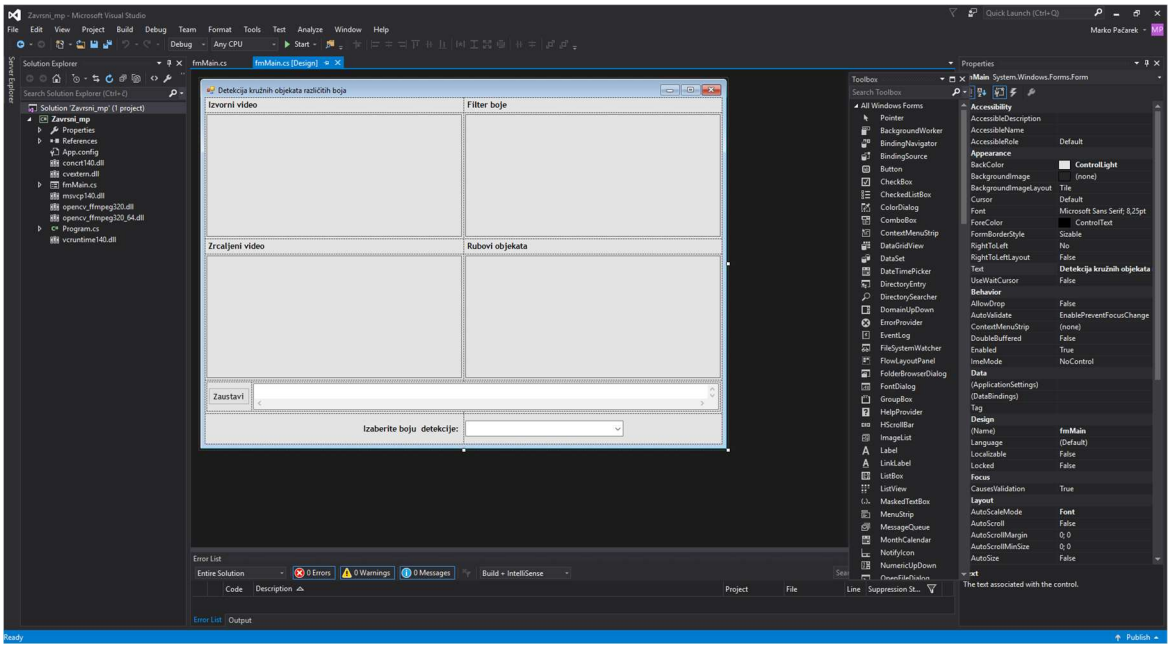
Na slici 2.1 prikazan je dio koda aplikacije. Slika prikazuje primjer sintakse C# jezika koju vrlo lako mogu razumjeti korisnici koji su koristili C ili C++, čak i ako se nikad nisu susreli sa C#. To dokazuje jednostavnost i laku prilagodljivost jeziku te opravdava tezu mnogih programera da je C# „najbolji programski jezik“.

```
13
14  using ...
30
31  namespace Zavrsni_mp
32  {
33      public partial class fmMain : Form
34      {
35          // member variables ///////////////////////////////////////////////////////////////////////////
36          VideoCapture capWebcam;
37          bool blnCapturingInProcess = false;
38          private readonly object Null;
39
40          // constructor ////////////////////////////////////////////////////////////////////////////////
41          public fmMain()
42          {
43              InitializeComponent();
44
45          }
46
47          private void fmMain_Load(object sender, EventArgs e)
48          {
49              try
50              {
51                  capWebcam = new VideoCapture();
52              }
53              catch (Exception ex)
54              {
55                  MessageBox.Show("unable to read from webcam, error: " + Environment.NewLine + Environment.NewLine +
56                                  ex.Message + Environment.NewLine + Environment.NewLine +
57                                  "exiting program");
58                  Environment.Exit(0);
59                  return;
60              }
61              Application.Idle += processFrameAndUpdateGUI;       // add process image function to the application's list of tasks
62              blnCapturingInProcess = true;
63          }
64          ///////////////////////////////////////////////////////////////////////////////////////////////
65          void processFrameAndUpdateGUI(object sender, EventArgs arg)
66          {
67              Mat imgOriginal;
68
69              imgOriginal = capWebcam.QueryFrame();
```

Sl. 2.1. *Primjer koda napisan u C# programskom jeziku*

2.2 Microsoft Visual Studio

Visual Studio je integrirano razvojno okruženje (ili IDE) napravljen od strane Microsofta u svrhu razvoja *desktop* aplikacija, aplikacija za Windows, Web stranica itd. Programski jezici koji se najčešće koriste u Visual Studio-u su C, C++, C# i Visual Basic. Visual Studio je praktičan alat jer podržava i dizajnerski pristup izradi programskog koda koji je prikazan na slici 2.2. To ubrzava proces kodiranja jer dijelove koda ne treba pisati, nego se programiranje može obavljati raspoređivanjem grafičkih objekta koji predstavljaju dijelove koda pri čemu je potrebno upisati samo neke osnovne značajke (engl. *Properties*) tih dijelova koda. [3]



Sl. 2.2. *Sučelje Microsoft Visual Studio okruženja*

Za izradu aplikacije korišten je Visual Studio 2015, koji sadrži gotovo sve mogućnosti kao i najnovija verzija, Visual Studio 2017. Starija verzija je odabrana za izradu rada zbog navike na njegovo grafičko sučelje i da ne bi došlo do problema sa kompatibilnosti, što se zna događati sa novom verzijom programa zbog sporog ažuriranja biblioteka i sl. Cijena Visual Studio-a 2017 Enterprise trenutno iznosi 2569,00 USD (nešto manje od 17 tisuća kuna), međutim studentima tehničkih fakulteta omogućene su besplatne licence za vrijeme trajanja studija. Postoje još „Proffesional“ i „Community“ verzije Visual Studio-a koje su znatno jeftinije od „Enterprise“ verzije, ali sadrže manje opcija i mogućnosti za rad sa njima. [4]

2.3 OpenCV / EmguCV

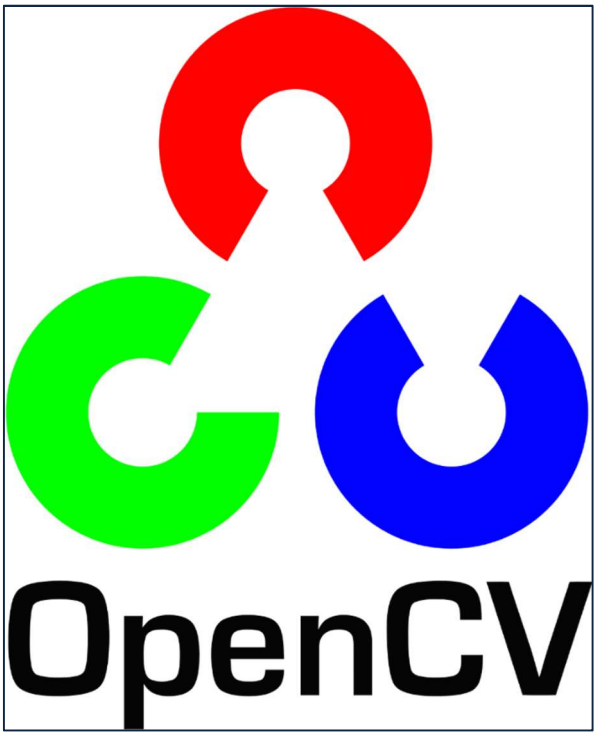
Radi zadatka izrade aplikacije koja uključuje područje računalnog vida (detekcija objekta, Web kamera) i odabira C# kao programskog jezika u izradi, potrebno je koristiti EmguCV biblioteke. EmguCV je posebna inačica poznatijeg skupa biblioteka OpenCV koja služi za pozivanje funkcija računalnog vida na jezicima koji podržavaju .NET platformu kao što su C#, VB, VC++, IronPython itd.

Sl. 2.3. *Logotip EmguCV-a* [5]

Računalni vid je područje koje obuhvaća metode za stjecanje, obrade, analiziranja i razumijevanja slike i općenito, strukturnih podataka iz realnog svijeta u svrhu dobivanja informacija u obliku brojeva ili simbola. Srodan je mnogim drugim područjima kao što su optika, automatika, robotika, umjetna inteligencija i sl. Koristi se u prometu, industriji, medicini, sigurnosti, svemirska istraživanja te vojne svrhe.[6]

OpenCV (engl. Open Source Computer Vision) skup biblioteka otvorenog koda koji je pokrenut od strane američke tvrtke Intel (Integrated Electronics Corporation) 1999. godine u svom istraživačkom centru u Rusiji. Biblioteke su višeplatformske (engl. cross-platform, multi-platform), mogu se koristiti u raznim aplikacijama, na raznim operacijskim sustavima. OpenCV služi raznim korisnicima da iskoriste i preoblikuju vlastiti kod. Biblioteka sadrži više od 2500 već gotovih optimiziranih algoritama. Algoritme možemo koristiti za pronalaženje, odnosno otkrivanje i prepoznavanje lica. Također se koristiti i za prepoznavanje objekata, praćenje objekata u pokretu itd.

Osnovna zadaća OpenCV-a pružanje je jednostavne platforme za rad sa računalnim vidom i omogućivanje jednostavnog i brzog razvoja jednostavnih i složenih algoritama a s time i aplikacija koje koriste računalni vid. Zbog kompliciranih algoritama koji rade sa velikim brojem operacija i podataka zahtjeva se optimiziran kod napisan na nižoj razini zbog veće brzine izvršavanja. Zbog toga su algoritmi OpenCV biblioteka orginalno pisani u C programskom jeziku. Zajednica OpenCV korisnika okuplja 50 tisuća ljudi koji su napravili oko 14 milijuna preuzimanja. Najčešći korisnici su tvrtke, istraživačke skupine, državna tijela, studenti itd. [7]



Sl. 2.4. *Logotip OpenCV-a* [8]

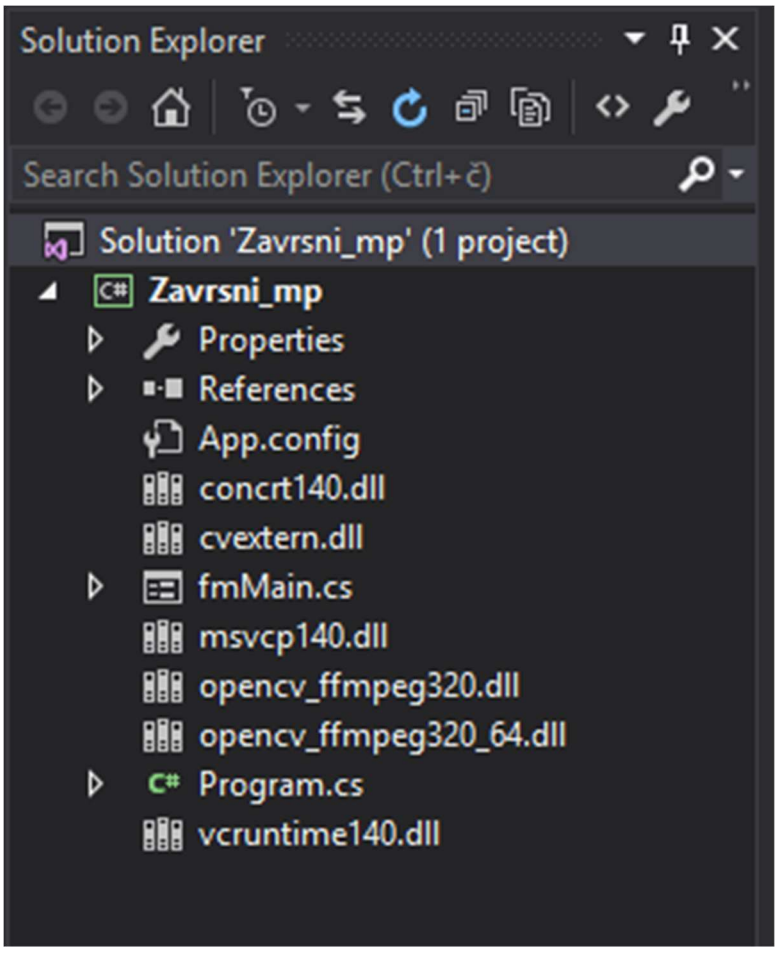
Način na koji se povezuju OpenCV/ EmguCV i C# opisan je na slijedećim stranicama u podnaslovu „3.1 Izrada aplikacije“ .

3. APLIKACIJA

3.1 Izrada aplikacije

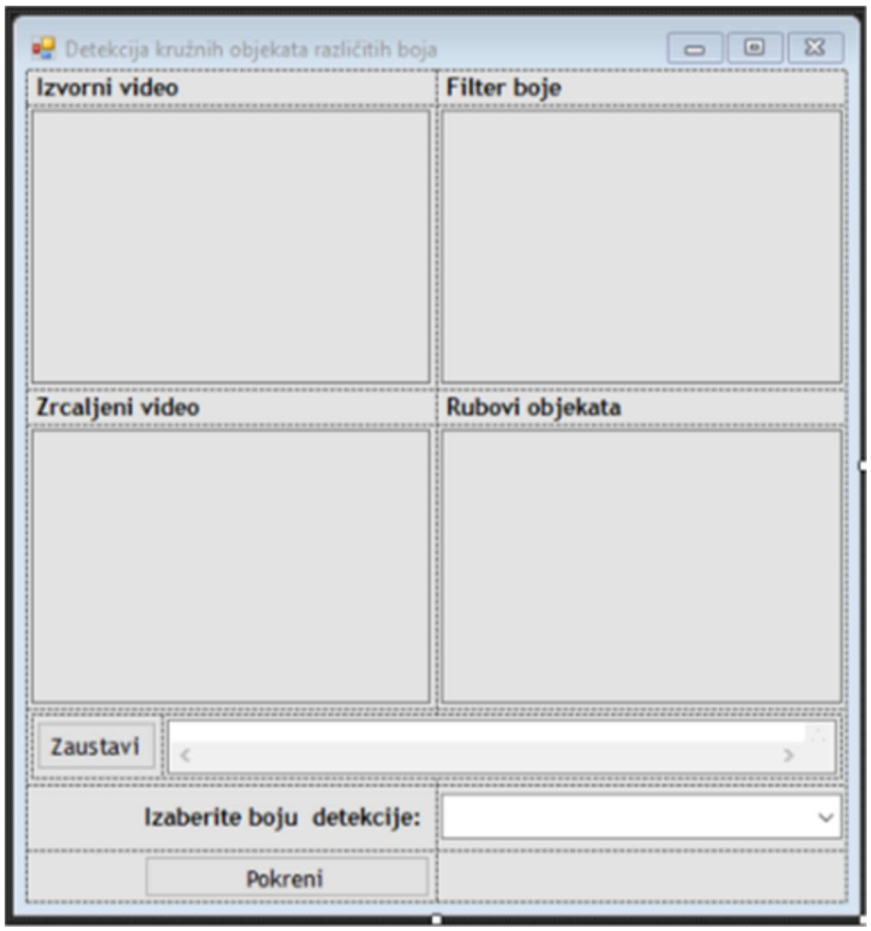
Kao što je prethodno spomenuto, aplikacija je pisana u C# programskom jeziku u Visual Studio razvojnom okruženju. Korištene su funkcije EmguCV/ OpenCV biblioteke. Kako bi mogli koristi EmguCV/ OpenCV biblioteke u Visual Studio-u, potrebno ih je najprije implementirati.

Za implementaciju potrebno je preuzeti najnovije ažuriranu EmguCV datoteku sa [9], te stvoriti vezu između EmguCV biblioteke, Visual Studio-a i korištenog operacijskog sustava. Tijekom izrade aplikacije korišten je Windows 10 operacijski sustav i nije bilo nikakvih problema što se tiče implementacije. Pri otvaranju samog projekta, u *Solution Explorer-u* (*Sl. 3.1.*), daju se uočiti implementirane datoteke, kao i ostale bitne datoteke projekta .



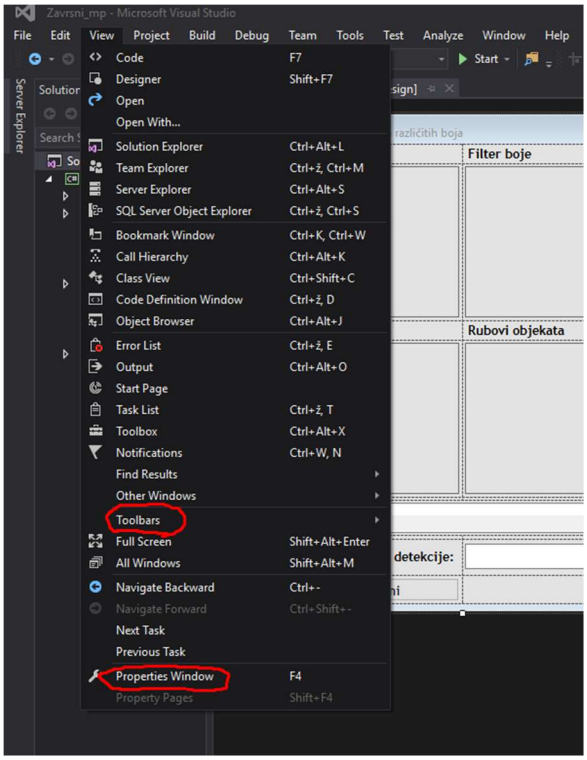
Sl. 3.1. *Najvažnije datoteke projekta*

Kod izrade većine C# aplikacija, najčešće se radi u dva različita prikaza. U *Design* prikazu (kojim definiramo izgled aplikacije i raspored vidljivih elemenata) i u *Source* prikazu (u kojem se nalazi kod koji definira aplikaciju). Prvo se u *Design* prikazu aplikacije pravi grafičko sučelje (*Sl. 3.2.*) koje nam predstavlja kako će krajnji korisnik vidjeti aplikaciju.

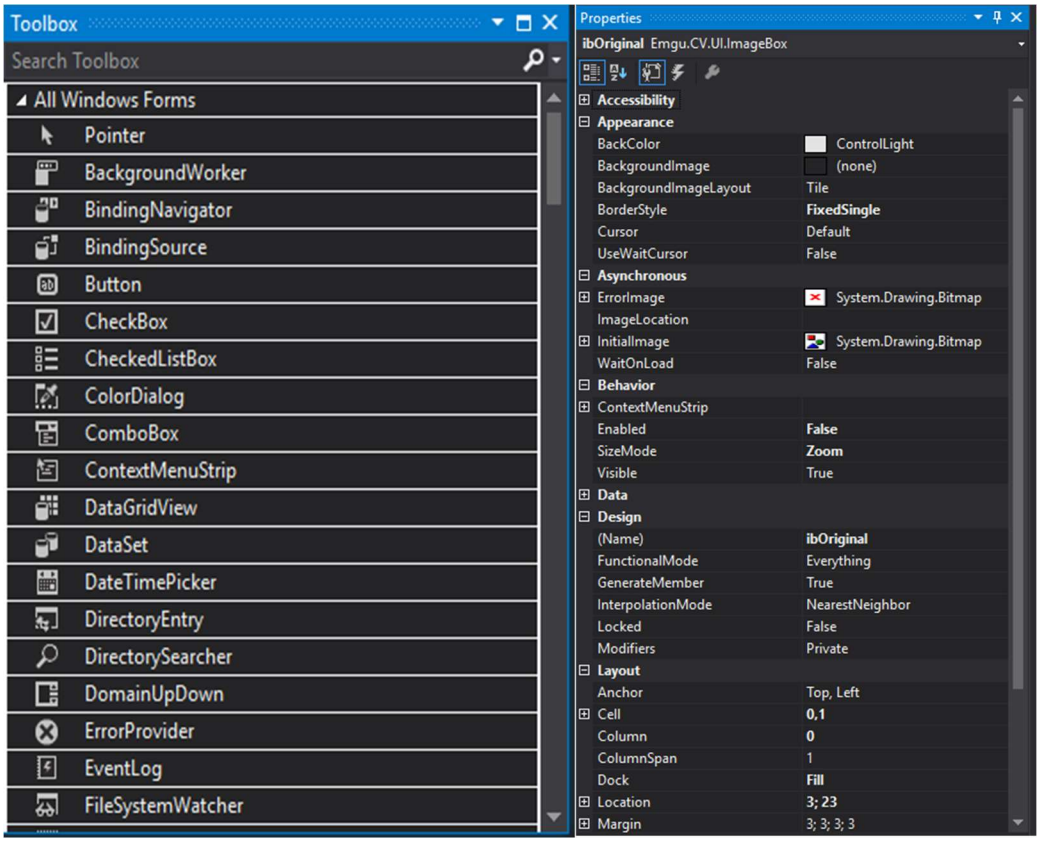


Sl. 3.2. *Izgled sučelja u Design prikazu*

Za izradu sučelja potrebno je (ako po *defaultu* nisu), uključiti alatne trake Toolbox i Properties. Na slici 3.3. prikazano je gdje se uključuju spomenute alatne trake, a na slici 3.4. može se vidjeti kako izgledaju uključene alatne trake. *Toolbox* služi za odabir dijelova od kojih će se sastojati sučelje (*Button, Combobox, Label*…), a *Properties* za definiranje njihovih svojstava (*Visible, Dock, Name…*).



Sl. 3.3. *Uključivanje alatnih traka Toolbars i Properties*



Sl. 3.4. *Toolbars i Properties*

Za kraj se piše programski kod u *Source* prikazu, a u kojem se nalaze biblioteke, deklaracije, funkcije, uvjeti itd. Dio programskog koda preuzet je sa GitHub-a i nadograđen je kako bi se dobila aplikacija tražena u zadatku završnog rada. Kod koji je preuzet je besplatan i svima dostupan na [10]. U aplikaciji se poziva mnoštvo EmguCV / OpenCV funkcija, kao što su: *Circle, CvtColor, Canny, Flip, GaussianBlur, Dilate, Erode* itd. Sve te funkcije imaju već unaprijed određene algoritme i tako olakšavaju programiranje aplikacija u području računalnog vida. Neki od ključnih dijelova programskog koda opisani su u nastavku rada.

```
if (comboBox1.Text == "crvena")
          CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(160, 155, 155)),
                 new ScalarArray(new MCvScalar(179, 255, 255)), imgThresh);
          else if (comboBox1.Text == "narančasta")
           CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(0, 155, 155)),
                 new ScalarArray(new MCvScalar(22, 255, 255)), imgThresh);
          else if (comboBox1.Text == "žuta")
           CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(22, 155, 155)),
                 new ScalarArray(new MCvScalar(38, 255, 255)), imgThresh);
          else if (comboBox1.Text == "zelena")
           CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(38, 155, 155)),
                 new ScalarArray(new MCvScalar(75, 255, 255)), imgThresh);
          else if (comboBox1.Text == "plava")
           CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(75, 155, 155)),
                 new ScalarArray(new MCvScalar(130, 255, 255)), imgThresh);
          else if (comboBox1.Text == "ljubičasta")
           CvInvoke.InRange(imgHSV, new ScalarArray(new MCvScalar(130, 155, 155)),
                       new ScalarArray(new MCvScalar(160, 255, 255)), imgThresh);
```

Programski kod 3.1. *Odabir boje (comboBox)*

Odabir boje u aplikaciji je realiziran pomoću padajućeg izbornika gdje je moguće odabrati samo jednu boju detekcije. Funkcija koja se koristi za filter određene boje zove se „CvInvoke.InRange“. Potrebno je u funkciji zadati gornju i donju vrijednost spektra boje u HSV (*hue, saturation, value*) formatu. Konkretno, u aplikaciji smo mijenjali samo *hue* parametar, dok smo ostala dva parametra ostavili nepromijenjena. Nešto više od filteru boja objašnjeno je u rezultatima aplikacije.

```
          Mat imgFlip = new Mat(imgOriginal.Size, DepthType.Cv8U, 1);
          CvInvoke.Flip(imgOriginal, imgFlip, 0);
          ibFlip.Image = imgFlip;
```

Programski kod 3.2. *Zrcaljenje slike (Flip)*

Za zrcaljenje slike korištena je gotova funkcija „CvInvoke.Flip“ sa kojom možemo odabrati želimo li video zrcaliti vertikalno ili okomito. Uz odabir načina zrcaljenja potrebno je odabrati koji video želimo zrcaliti i u koju varijablu želimo spremiti zrcaljeni video.

```
            Mat imgGrayscale = new Mat(imgOriginal.Size, DepthType.Cv8U, 1);
            Mat imgBlurred = new Mat(imgOriginal.Size, DepthType.Cv8U, 1);
            Mat imgCanny = new Mat(imgOriginal.Size, DepthType.Cv8U, 1);

            CvInvoke.CvtColor(imgOriginal, imgGrayscale, ColorConversion.Bgr2Gray);

            CvInvoke.GaussianBlur(imgGrayscale, imgBlurred, new Size(5, 5), 1.5);

            CvInvoke.Canny(imgBlurred, imgCanny, 100, 200);

            ibOriginal.Image = imgOriginal;
            ibCanny.Image = imgCanny;
```

Programski kod 3.3. *Rubovi objekta (Canny)*

Funkcija za detekciju rubova „Canny“ dobila je ime po čovjeku (J. Canny) koji je preradio Laplaceovu metodu za pronalazak rubova. Funkcija radi po principu da ako *piksel* ima gradijent veći od gornjeg praga, prihvaćen je kao dio ruba, a ako je *piksel* ispod donjeg praga, odbacuje se mogućnost da je dio ruba. Ako je *piksel* između gornje i donje vrijednosti, bit će prihvaćen samo ako spojen (neposredno blizu). Algoritam je toliko dobar koliko je i izvorni video. Veliki utjecaj ima osvjetljenje, tj. Sjene koje čine veliki problem.

```
        void processFrameAndUpdateGUI(object sender, EventArgs arg)
        {
            Mat imgOriginal;

            imgOriginal = capWebcam.QueryFrame();


            if (imgOriginal == null)
            {
                MessageBox.Show("unable to read from webcam" + Environment.NewLine +
                                Environment.NewLine + "exiting program");
                Environment.Exit(0);
                return;
            }
```

Programski kod 3.4. *Učitavanje video prikaza sa Web kamere*

Video sa Web kamere se učitava u varijablu koja mora biti deklarirana kao „Mat“ klasa. Učitavanje se izvršava „capWebcam.QueryFrame“ funkcijom i nakon toga je potrebno provjeriti je li učitavanje slike uspješno izvršeno. Provjera se izvršava postavljajući uvjet da je varijabla „Mat“ klase jednaka „*null*“ , odnosno da je prazna.

```
                if (txtXYRadius.Text != "")
                {
                    txtXYRadius.AppendText(Environment.NewLine);
                }

                txtXYRadius.AppendText("ball position x = " +
                    circle.Center.X.ToString().PadLeft(4) +
                    ", y = " + circle.Center.Y.ToString().PadLeft(4) +
                    ", radius = " + circle.Radius.ToString("###.000").PadLeft(7));
                txtXYRadius.ScrollToCaret();
```

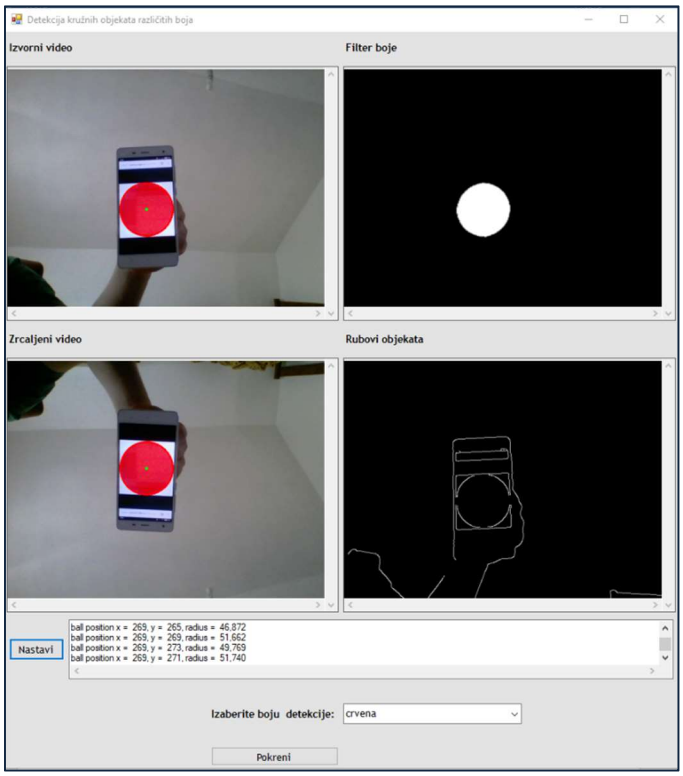
Programski kod 3.5. *Ispis koordinata središta i polumjera kružnog objekta*

Ovaj dio koda služi za ispis koordinata središta detektiranog objekta i njegovog polumjera. Radi tako što prebacuje ispis na kraj već ispisanog sadržaja ako ga je bilo. Ispisuje se lokacija detektiranog objekta i veličina njegovog polumjera. Također je omogućeno automatsko *scrollanje* ispisanog sadržaja tako da pokazuje na zadnje ispisani tekst.

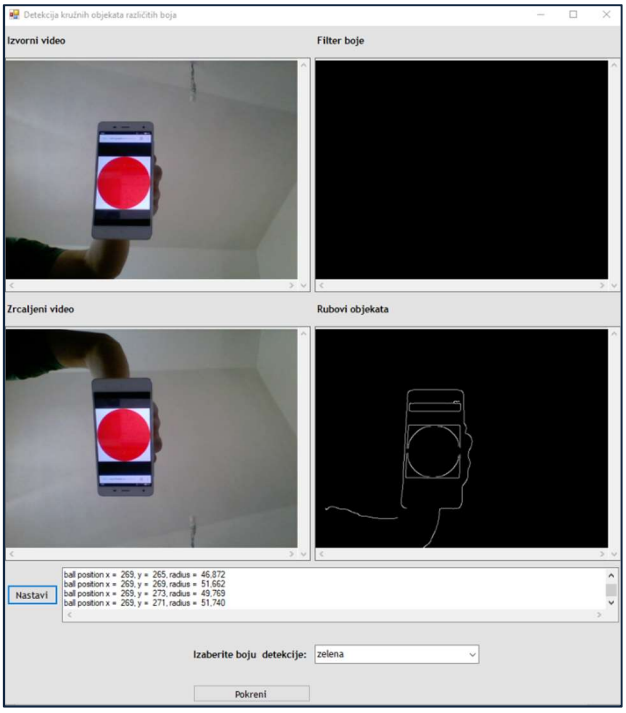
3.2 Rezultati aplikacije

Nakon pokretanja aplikacija i odabira tražene boje, aplikacija detektira kružne objekte i ispisuje njihove koordinate kao i duljinu polumjera kružnice koja opisuje taj objekt. Središte objekta (zelena točka) i kružnica (crvena kružnica) koja ga opisuje također su i grafički označeni na izvornom i zrcaljenom videu. Postoji i filter boje koji traženu boju prikazuje bijelom, a sve ostale boje crnom radi lakšeg uočavanja predmeta. Prikaz rubova objekta isto pomaže kod detekcije tako što algoritam za traženje kružnica provjeri za sve rubove jesu li kužnog oblika.

Slika 3.5. detekciju kruga crvene boje. Aplikacija ispravno detektira objekt, označuje središte i opisanu kružnicu, te ispisuje koordinate središta i polumjer. To nije slučaj i na slici 3.6. gdje je za boju traženog objekta odabrana „zelena“. Aplikacija ne pronalazi kružni objekt zelene boje na videu tako da ništa ne ispisuje, niti prikazuje.

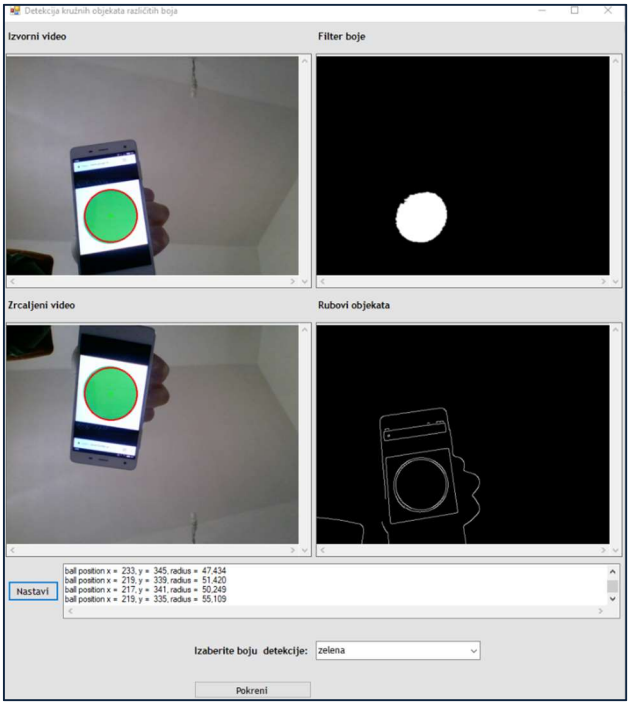


Sl. 3.5. *Detekcija kružnog oblika crvene boje uz odabir crvene boje*

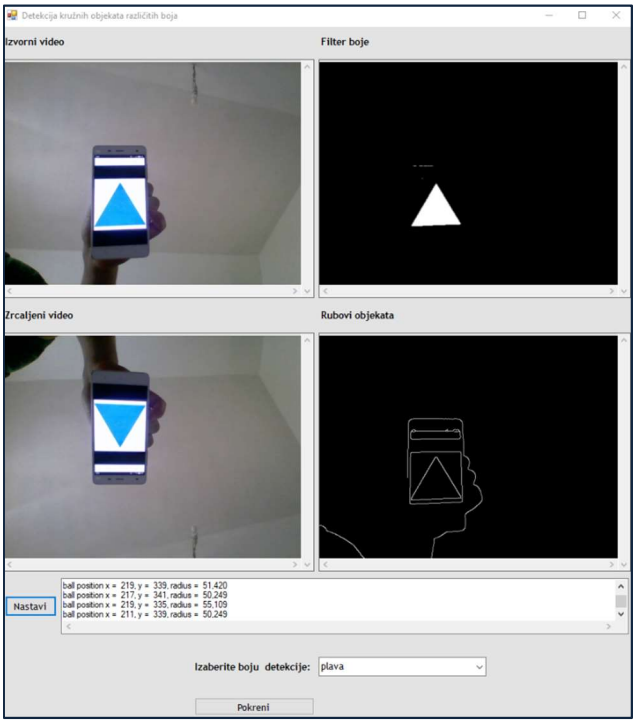


Sl. 3.6. *Detekcija kružnog oblika crvene boje uz odabir zelene boje*

U slijedećem primjeru aplikacija ispravno pronalazi kružni objekt zelene boje i izvršava sve što je potrebno (*Sl. 3.7.*). Kada je prikazan neki ne kružni objekt odabrane boje, tada aplikacija samo prikazuje tu boju na filteru boja (bijelom bojom). Tada aplikacija neće pratiti objekt niti označavati središte jer objekt nije kružnog oblika (*Sl. 3.8.*).



Sl. 3.7. *Detekcija kružnog objekta zelene boje uz odabir zelene boje*



Sl. 3.8. *Detekcija ne kružnog objekta plave boje uz odabir plave boje*

Princip rada filtera boje najbolje je prikazan slikom 3.9. gdje smo provjerili filter za svaku od šest elementarnih boja na Ostwaldovom krugu boja. Spektar boje zadajemo u HSV formatu u InRange funkciji. Konkretno, za crvenu boju potrebno je podesiti matricu tona boja tako da je minimalna vrijednost 160, a maksimalna 179 . Te vrijednosti ovise o nijansi žute boje, osvjetljenju i sličnim faktorima pa se određuju eksperimentalno. Vrijednosti koje su korištene prilikom izrade aplikacije su :

Narančasta: 0 – 22

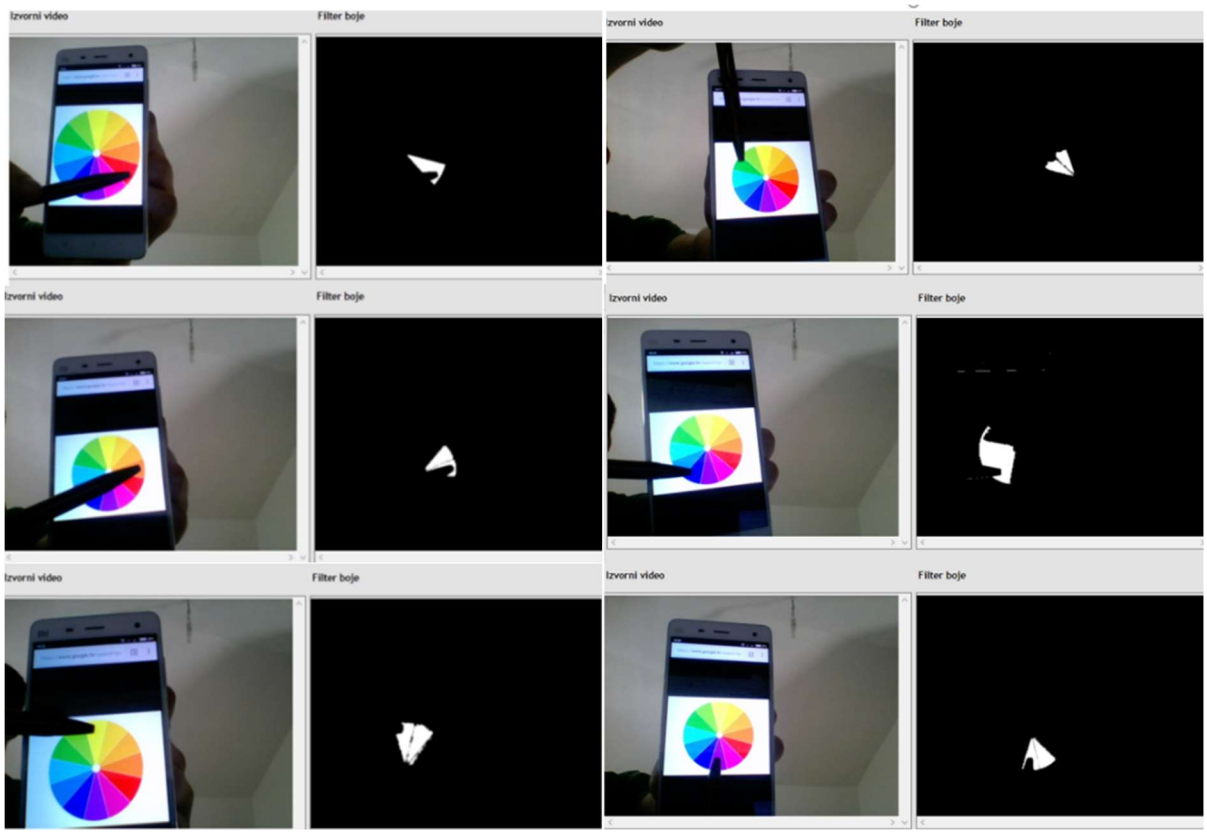
Žuta: 22 – 38

Zelena: 38 – 78

Plava: 75 – 130

Ljubičasta: 130 – 160

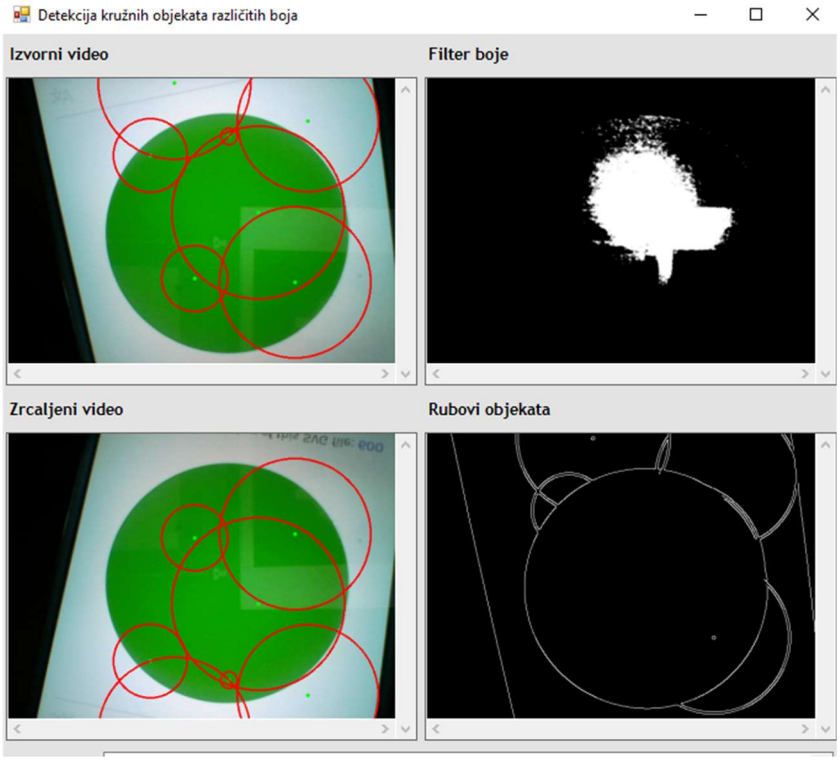
Crvena: 165 - 179



Sl. 3.9. *Filter boje*

3.3 Problemi i izazovi

Postoje razni problemi koji se javljaju zbog male rezolucije kamere, šumova, osvjetljenja, sjene, kuta iz kojeg snimamo objekt itd. Glavni problem prepoznavanja objekta u ovoj aplikaciji riješen je tako što se prepoznavanje temelji na kružnom obliku i boji. Ukoliko bi objekt bio druge boje od zadane, program taj objekt ne bi prepoznao kao objekt. Taj problem bi se mogao riješiti korištenjem infracrvene kamere koja nam daje uvid u udaljenost svakog pojedinog *piksela* od kamere. Međutim, takve kamere su sklone šumu i nepouzdane su za predmete koji su na velikoj udaljenosti od kamere.



Sl. 3.10. *Krivi rezultati kod velikih objekta*

Problem je i veličina objekta. Prevelike objekte aplikacija detektira kao više manjih pa nam prikazuje krive rezultate kao što vidimo na (*Sl. 3.10.)*. Male objekte također krivo detektira, pa je potrebno postaviti gornju i donju granicu za veličinu polumjera. Granice se postavljaju u funkciji „HoughCircles“ i to tako da za granice stavimo najmanje i najveće očekivane polumjere kružnica koje očekujemo.

Na loše osvjetljenje, sjenu, šumove i krivi kut slikanja možemo utjecati tako da ih pokušamo svesti na minimum. Uvelike će pridonijeti i korištenje kvalitetnije kamere, za razliku od *Web* kamere sa 0.3 megapiksela koja je korištena za testiranje aplikacije

4. ZAKLJUČAK

Aplikacija za detekciju objekta odabrane boje u stvarnom vremenu pomoću Web kamere je uspješno izrađena. Ovim seminarom prikazane su tehnologije i alati potrebni za izradu, kao i sam postupak izrade. Aplikacija sadrži sve potrebne elemente i izvršava ciljeve zadane u zadatku završnog rada. Naravno, ova aplikacija, kao i svaka druga, ne radi idealno. Postoje problemi koji se mogu ispraviti ažuriranjem aplikacije, ali postoje i oni na koje ne možemo utjecati. I jedni i drugi problemi detaljno su opisani u razradi završnog rada.

Postavlja se pitanje na koji bi se način aplikacija mogla unaprijediti. Jedno od rješenja je omogućiti korisniku odabir drugih oblika, a ne samo kružnih objekta. Druga opcija je povećati izbor boja predmeta kojeg detektiramo pa da se umjesto trenutnih šest, može odabrati gotovo svaka nijansa boje koja postoji. To bi rezultiralo boljom i preciznijom detekcijom koja bi svoju primjenu mogla pronaći u praksi. Aplikacije ovakvog tipa imaju budućnost u području kao što je sport zbog toga što se teži k tome da se izuzme pogreška ljudskog faktora u obliku suđenja. Aplikacije za detekciju bi se mogle koristiti i u prometu, u svrhu prepoznavanja znakova i registracijskih oznaka ili za izbjegavanje prepreka u autonomnim vozilima. Postoji neograničen broj ideja za što bi se sve mogao koristiti računalni vid pa je poznavanje biblioteke OpenCV jako korisno.

LITERATURA

[1] Sharp, John. Microsoft Visual C# 2012: Step by Step, Microsoft, 2012

[2] FERIT Osijek, Objektno orijentirano programiranje:
Auditorne vježbe 5 - Uvod u C#, str.1, 2017.

[3] Microsoft Visual Studio, dostupno na:
https://www.visualstudio.com/ [24. lipnja 2017.]

[4] Microsoft Visual Studio cijena, dostupno na:
https://www.microsoft.com/ [24. lipnja 2017.]

[5] EmguCV logotip, dostupno na:
https://sourceforge.net/projects/emgucv/ [24. lipnja 2017.]

[6] Računalni vid, Wikipedia, 2017., dostupno na:
https://hr.wikipedia.org/wiki/Računalni_vid [24. lipnja 2017.]

[7] OpenCV , dostupno na:
http://opencv.org/ [24. lipnja 2017.]

[8] OpenCV logotip, Wikipedia, 2017., dostupno na:
https://en.wikipedia.org/wiki/OpenCV [24. lipnja 2017.]

[9] EmguCV biblioteka, dostupno na:
http://www.emgu.com/wiki/index.php/Download_And_Installation [24. lipnja 2017.]

[10] GitHub kod, dostupno na:
https://github.com/MicrocontrollersAndMore [24. lipnja 2017.]

SAŽETAK

Zadatak ovog rada bio je napraviti desktop aplikaciju koja će omogućiti detekciju objekta odabrane boje u stvarnom vremenu pomoću Web kamere. Za izradu aplikacije korišten je programski jezik C#, skup biblioteka EmguCV/OpenCV koje služe za pozivanje funkcija koje obuhvaćaju područja računalnog vida te razvojno okruženje Microsoft Visual Studio 2015. Najprije su implementirane biblioteke EmguCV-a u razvojno okruženje kako bi se moglo pristupiti funkcijama računalnog vida. Nakon toga je izrađeno sučelje aplikacije kako bi korisnik imao prikaz detekcije u stvarnom vremenu. Ostatak je realiziran pisanjem C# koda i pozivanjem EmguCV funkcija. Kao rezultat dobivena je desktop aplikacija kojom je moguće detektirati objekte kružnog oblika i pratiti njihovo gibanje. Također je moguće odabrati jednu od šest elementarnih boja koju želimo da aplikacija detektira i prikaže na zaslonu u stvarnom vremenu.

Ključne riječi: C#, OpenCV, EmguCV, Microsoft Visual Studio, računalni vid, detekcija objekta

ABSTRACT

Object detection using Web camera and OpenCV

The goal of this paper was to make a desktop application that enables a real-time detection of an object of a chosen color using the Web cam. For the realization of the application, a C# programming language was used, as well as the collection of EmguCV/OpenCV libraries (used for the calling of functions that refer to the computer vision) and the Microsoft Visual Studio 2015 developing environment. Firstly, the EmguCV libraries were implemented into the developing environment, in order to have access to the functions of computer vision. Afterwards, the user interface of the application was built, so that the user could have access to the display of the detection in real time. The rest was accomplished by writing the C# code and using the EmguCV functions. The result was a desktop application that enables the detection of circular objects and tracks their motion. Also, it is possible to select any of the six elementary colors that one wants the application to detect and display on the screen in real time.

Keywords: C#, OpenCV, EmguCV, Microsoft Visual Studio, computer vision, object detection

ŽIVOTOPIS

Marko Pačarek rođen je 11. travnja 1996. godine u Požegi, Hrvatska. Živi u Kutjevu, gdje 2002. godine počinje s osnovnoškolskim obrazovanjem u OŠ Zdenka Turkovića. Tijekom osnovnoškolskog obrazovanja sudjeluje na brojnim županijskim natjecanjima iz područja matematike, fizike, geografije i kemije. Nakon završetka osnovne škole, 2010. godine upisuje se u Tehničku školu u Požegi, smjer Tehničar za računalstvo. Četiri godine poslije maturira sa odličnim uspjehom i upisuje preddiplomski studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija, koji i dalje pohađa. Posjeduje osnovna znanja iz programskih jezika C, C++, C#, PHP ,SQL i alata kao što su HTML i CSS.

_____

Marko Pačarek