Uklanjanje objekata iz fotografija

Volarev, Lazar

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

Sveučilišni studij

UKLANJANJE OBJEKATA IZ FOTOGRAFIJE

Završni rad

Lazar Volarev

Osijek, 2022. godina.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 20.09.2022.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Lazar Volarev
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R 4439, 22.07.2019.
OIB Pristupnika:	39265617680
Mentor:	izv. prof. dr. sc. Časlav Livada
Sumentor:	,
Sumentor iz tvrtke:	
Naslov završnog rada:	Uklanjanje objekata iz fotografija
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Zadatak završnog rad:	U radu je potrebno korištenjem OpenCV biblioteke maknuti objekte iz fotografija te provjeriti mogućnosti skripte na složenost fotografije (objekti, ljudi, ...). U teorijskom dijelu potrebno je opisati rad gore navedene skripte . Tema rezervirana za: Lazar Volarev
Prijedlog ocjene završnog rada:	Dobar (3)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 1 bod/boda Jasnoća pismenog izražavanja: 1 bod/boda Razina samostalnosti: 2 razina
Datum prijedloga ocjene od strane mentora:	20.09.2022.
Datum potvrde ocjene od strane Odbora:	21.09.2022.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 23.03.2023.

Ime i prezime studenta:	Lazar Volarev
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R 4439, 22.07.2019.
Turnitin podudaranje [%]:	12

Ovom izjavom izjavljujem da je rad pod nazivom**: Uklanjanje objekata iz fotografija**

izrađen pod vodstvom mentora izv. prof. dr. sc. Časlav Livada

i sumentora ,

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

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. SLIČNE APLIKACIJE 2

2.1. Retouch 2

2.2. Adobe Photoshop Fix 2

2.3. Snapseed 3

2.4. Pixlr – Photo Editor 3

2.5. PhotoDirector Essential 4

3. RAZVOJNI ALATI I TEHNOLOGIJE 5

3.1. Python 5

3.2. OpenCV 5

3.3. Visual Studio Code 6

4. IZRADA APLIKACIJE 7

4.1. DetectAndRemove 7

4.2. Main 10

5. KORIŠTENJE APLIKACIJE 13

6. ZAKLJUČAK 20

7. SAŽETAK 21

8. ABSTRACT 22

9. LITERATURA 23

10. IZVOR SLIKA 24

11. PRILOZI 25

1. UVOD

Dana sve više ljudi gradi svoju karijeru preko društvenih mreža, te je sve veća potreba za što kvalitetnijim sadržajem, pa između ostalog i za slikama. Većina želi da njihove slike izgledaju savršeno, bez ijedne mane, te se stoga odlučuju na uređivanje slika putem raznih aplikacija ili editora. Na tržištu već postoji pregršt takvih aplikacija koje savjetuju korisnike kako da urede sliku da bude savršena, ali da pritom ostane i realna.

Cilj ovog završnog rada je istražiti postojeća rješenja, te iz njih izvući i napraviti što bolju aplikaciju. Ova aplikacija će omogućiti korisniku prepoznavanje objekata na bilo kojoj slici, te uklanjanje željenih.

U drugom poglavlju su opisane već postojeće aplikacije koje su dostupne na tržištu. Treće poglavlje prikazuje programske jezike i tehnologije koje su korištene tijekom stvaranja aplikacije. Četvrto poglavlje sadrži najbitnije dijelove programskog koda same aplikacije, dok peto poglavlje prikazuje testiranje aplikacije s određenim fotografijama, gdje se ispituje i analizira točnost aplikacije.

1.1. Zadatak završnog rada

U teorijskom dijelu rada potrebno je opisati način kreiranja aplikacije, to jest rad skripte. Dok je u programskom dijelu potrebno kreirati aplikaciju koja bi u određenoj fotografiji prepoznala sve objekte koji se na njoj nalaze, te željene objekte uklonila. Mjesto objekta je potrebno popuniti sličnom okolinom. Na kraju je potrebno provjeriti složenost fotografije na broj objekata.

2. SLIČNE APLIKACIJE

U današnje vrijeme dostupno je puno aplikacija za uređivanje fotografija, te brisanja raznih objekata s njih. Štoviše jako puno ljudi koriste takve aplikacije, počevši od raznih filtera na društvenim mrežama koji su dostupni svima, pa sve do ljudi koji se profesionalno bave uređivanjem fotografija, te brisanjem određenih stvari iz njih. Potreba za ovakvim aplikacijama je nastala zato što ljudi žele savršene fotografije, te su ih spremni uređivati satima i satima.

2.1. Retouch

Prema [1], *Retouch* je aplikacija koja nudi mogućnost uklanjanja neželjenih objekata s fotografije. No, osim toga, s njom se može očistiti tekst ili logo s fotografije i retuširati fotografija. Još neke mogućnosti su: zamjena pozadine i kloniranje objekata na istu fotografiju. Ova aplikacija je dostupna za *Android* uređaje. Na slici 2.1 je prikazan logo *Retoucha*.

Slika 2.1 *Retouch* logo [1]

2.2. Adobe Photoshop Fix

Prema [2], to je aplikacija koju je razvio *Adobe Systems Incorporated*. Bila je dostupna za *Android*, te *iOS* korisnike, ali nakon 21. srpnja 2021. godine je uklonjena za preuzimanje na *App Storeu*. Vrlo je popularna, ima više od 10 milijuna preuzimanja na *Google Playu*. Jedna od dodatnih mogućnosti ove aplikacije jest da se fotografije koje su retuširane mogu izvesti u *Adobe Photoshop desktop* verziji. Slika 2.2 prikazuje logo aplikacije.

Slika 2.2. *Adobe Photoshop Fix* [2]

2.3. Snapseed

U slučaju da se briše osoba ili predmet, ova aplikacija je jako slična *Adobe Photoshop Fixu*. Prema [3], kod uklanjanja objekta dodaje se pozadina koja je jednaka okruženju, tako da fotografija nema prazninu. *Snapseed* podržava *JPG* i *RAW* datoteke, što je sasvim dovoljno, jer većina fotografija je *JPG* datoteka. Ovu aplikaciju razvio je *Google*, te ona ima preko 100 milijuna preuzimanja na *Google Playu*. Zahtjeva verziju *Androida* 4.0 ili noviju. Aplikacija je besplatna, te je za razliku od *Adobea* dostupna na *App Storeu*. Logo aplikacije vidljiv je na slici 2.3.

Slika 2.3. *Snapseed* [3]

2.4. Pixlr – Photo Editor

Prema [4], *Pixlr* je uređivač fotografija, dostupan za korisnike pametnih telefona koji koriste *Android* ili *iOS* operacijski sustav. Besplatan je, te sadrži preko 2 milijuna efekata, prekrivača i filtera. Prije se ova aplikacija zvala *Pixlr Express*. Za uređivanje fotografija nije potreban korisnički račun, nego se jednostavno preuzme aplikacija i samo se počne uređivati. Sučelje je vidljivo na slici 2.4. Prosječna ocjena na *Google Playu* je 4.2 i ima 1.21 milijuna recenzija od 50 milijuna preuzimanja.

Slika 2.4. Primjer sučelja u *Pixlru* [4]

2.5. PhotoDirector Essential

Prema [5], *PhotoDirector Essential* je nagrađivani program za uređivanje fotografija. To je besplatna verzija koju je razvio *CyberLink*. *PhotoDirector* je prvenstveno dostupan za korištenje na *Windows* i *MacOs* operativnim sustavima, ali zbog velikog korištenja, razvile su se i aplikacije za *iOS* i *Android*. Sučelje aplikacije jasno se vidi na slici 2.5.

Slika 2.5. Sučelje *PhotoDirector Essential* [5]

3. RAZVOJNI ALATI I TEHNOLOGIJE

3.1. Python

Prema [6], *Python* je programski jezik visoke razine koji je stvoren 1991. godine. Kreator *Pythona* je Nizozemac Guido van Rossum, koji mu je dao ime po *Monty Pythonu*. Ima određene sličnosti s ostalim programskim jezicima, a to su *Ruby, Perl, Smalltalk*… Vrlo je popularan, te je jedan od najkorištenijih programskih jezika na svijetu. Logo Pythona dostupan je na slici 3.1. Postoje tri verzije, najstariji *Python* 1.0 objavljen je 1994. godine, iduća verzija *Python* 2.0 svjetlost dana je ugledao 2000. godine, dok je najnovija verzija *Python* 3.0 izašao 2008. godine.

Slika 3.1. Logotip *Pythona* [6]

3.2. OpenCV

Prema [7], *OpenCV* (*Open Source Computer Vision*) je biblioteka za strojno učenje (engl. *machine learning*) i računalni vid otvorenog koda. Originalno ju je razvila tvrtka *Intel* 2000. godine.

OpenCV uključuje:

- Detekciju objekta
- Detekciju pokreta
- Prepoznavanje lica
- Segmentaciju
- Mobilnu robotiku
- 2D i 3D alate
- Interakciju čovjeka i računala
- Prepoznavanje geste
- Praćenje pokreta
- Proširenu stvarnost

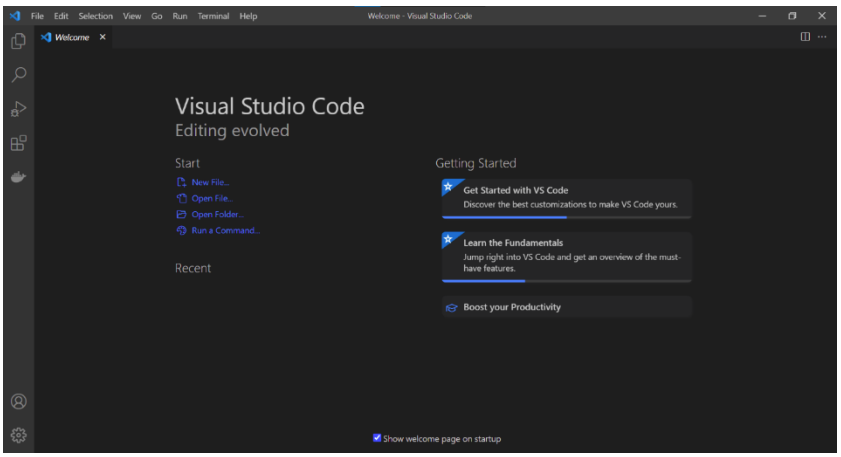
OpenCV je napisan u programskom jeziku *C++*, ali je povezan i sa *Javom* i *Pythonom*. Znak se vidi na slici 3.2. Dostupan je na *Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD,*

Android, iOS, Maemo te *Blackberry* 10 operativnim sustavima. Službena stranica *OpenCV-a* je „*opencv.org*“.

Slika 3.2. Logotip *OpenCV-a* [7]

3.3. Visual Studio Code

Prema [8], *Visual Studio Code* je integrirano razvojno okruženje (engl. *Integrated development environment IDE*) koje je razvila tvrtka *Microsoft* za *Windows, Linux i macOS*. To je editor otvorenog koda. U anketi za razvojne programere 2021. godine koju je provodio *Stack Overflow, VS Code* je rangiran kao najpopularniji alat za razvojno okruženje, te je 70% ispitanika izjavilo da ga koristi. Dostupan je za 14 programskih jezika, a neki koje podržava su *C, C++, C#, Java, Python, JavaScript, Fortran, Go*,… Prvi put je izdan 2015. godine. Izgled *Visual Studio Coda* vidljiv je na slici 3.3. Napisan je u *TypeScript, JavaScript, HTML (HyperText Markup Language)* i *CSS* jezicima.



Slika 3.3. *VS Code* 1.58 verzija na *Windowsu 10* [8]

4. IZRADA APLIKACIJE

4.1. DetectAndRemove

Prvo je kreirana *Python* datoteka „*DetectAndRemove*“ koja sadrži sve što je potrebno za detekciju objekata, te njihovo uklanjanje. Koristile su se još i datoteke „*frozen_inference_graph.pb*“, „*ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt*“, te datoteka „*coco.names*“. Ove datoteke nalaze se u prilozima.

Prvi korak je učitavanje biblioteka koje ćemo koristiti u pisanju programa. Slika 4.1. prikazuje biblioteke koje su bile neophodne za rad, a slika 4.2. pokazuje funkciju koja pohranjuje imena objekata koja se nalaze u datoteci „*coco.names*“ u obliku liste *stringova*.

```
1  import cv2
2  import numpy
```

Slika 4.1. Učitavanje potrebnih biblioteka [12]

Učitane biblioteke.

```
6  def getClasses():
7      with open('Object_Detection_Files/coco.names', 'rt') as file:
8          
9          classes = file.read().rstrip('\n').split('\n')
10     return classes
```

Slika 4.2. Funkcija *getClasses()* [12]

Na slici 4.3. se može vidjeti povratna konfiguracija obrade slike koristeći mobilni *SSD* model.

Prema [9], *SSD* model je mreža namijenjena otkrivanju objekata. Ovaj model je implementiran pomoću *Caffe** okvira.

```
14 def getConfiguration():
15     net = cv2.dnn_DetectionModel('Object_Detection_Files/frozen_inference_graph.pb',
16                                  'Object_Detection_Files/ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt')
17     net.setInputSize(320,320)
18     net.setInputScale(1.0/ 127.5)
19     net.setInputMean((127.5, 127.5, 127.5))
20     net.setInputSwapRB(True)
21 
22     return net
23 
```

Slika 4.3. Funkcija *getConfiguration()* [12]

Na slici 4.4 se vidi jedna od bitnijih funkcija. Ova funkcija detektira objekte unutar slike i vraća sliku s obilježenim objektima.

```
25  def detectObjects(imageP, net, classes):
26      img = cv2.imread(imageP)
27      displayImg = cv2.imread(imageP)
28
29
30      img = cv2.resize(img, [600, 600])
31      displayImg = cv2.resize(img, [600, 600])
32
33
34      classIDs, confidences, boxes = net.detect(img, confThreshold = 0.5)
35
36
37      boxes = list(boxes)
38      confidences = list(map(float, confidences))
39
40
41      boxesToOutput = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)
42
43
44      for i in boxesToOutput:
45          box = boxes[i]
46          x, y, w, h = box[0], box[1], box[2], box[3]
47
48
49          cv2.rectangle(displayImg, (x, y), (x + w, y + h), color = (0, 0, 255))
50
51
52          cv2.putText(displayImg, str(i + 1), (x + 1, y + 20),
53                      cv2.FONT_ITALIC, 0.5, (0, 255, 0), 2)
54      return [displayImg, img, boxes, boxesToOutput[-1]]
```

Slika 4.4. Funkcija *detectObjects()* [12]

S ovim dijelom koda koji se nalazi na slici 4.5. postiže se postavljanje slike na dimenzije 600x600.

```
30      img = cv2.resize(img, [600, 600])
31      displayImg = cv2.resize(img, [600, 600])
32
```

Slika 4.5. Stavljanje novih dimenzija [12]

Pokretanje detekcije i spremanje ID-ova objekata, povjerljivosti i graničnih okvira vidljivo je na slici 4.6.

```
34      classIDs, confidences, boxes = net.detect(img, confThreshold = 0.5)
```

Slika 4.6. Spremanje ID-ova [12]

Ovaj dio koda koji je prikazan na slici 4.7. služi za konvertiranje u liste.

```
37      boxes = list(boxes)
38      confidences = list(map(float, confidences))
```

Slika 4.7. Konvertiranje [12]

Korištenjem *NMS-a (Non Maximum Suppression*) uklanja se okvir s nižim pragovima, što je predstavljeno na slici 4.8.

```
41      boxesToOutput = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)
```

Slika 4.8. *NMS* [12]

For petljom se postiže izlaz graničnih okvira bez preklapanja, kao što se vidi na slici 4.9.

```
    for i in boxesToOutput:
        box = boxes[i]
        x, y, w, h = box[0], box[1], box[2], box[3]


        cv2.rectangle(displayImg, (x, y), (x + w, y + h), color = (0, 0, 255))

        cv2.putText(displayImg, str(i + 1), (x + 1, y + 20),
                    cv2.FONT_ITALIC, 0.5, (0, 255, 0), 2)
    return [displayImg, img, boxes, boxesToOutput[-1]]
```

Slika 4.9. *For* petlja [12]

Na slici 4.10. se vidi crtanje graničnih okvira na slici s pozicija spremljenih u x, y. Prema [10], funkcija *rectangle()* se koristi za crtanje pravokutnika na bilo kojoj slici.

```
        cv2.rectangle(displayImg, (x, y), (x + w, y + h), color = (0, 0, 255))
```

Slika 4.10. Crtanje okvira [12]

Završetak *for* petlje, te cijele funkcije *detectObjects()* koja vraća sliku sa detektiranim objektima vidi se na slici 4.11.

```
        cv2.putText(displayImg, str(i + 1), (x + 1, y + 20),
                    cv2.FONT_ITALIC, 0.5, (0, 255, 0), 2)
    return [displayImg, img, boxes, boxesToOutput[-1]]

```

Slika 4.11. Kraj funkcije *detectObjects()* [12]

Slika 4.12. pokazuje funkciju koja koristeći podatke iz funkcije *detectObjects()* stvara masku i briše odabrane objekte.

```
def deleteSelectedObjects(selectedObjects, boxes, img):

    mask = numpy.zeros_like(img)
    for i in selectedObjects:
        box = boxes[i - 1]
        x, y, w, h = box[0], box[1], box[2], box[3]


        mask[y:y + h + 1, x: x + w + 1] = 255


    mask = cv2.cvtColor(mask, cv2.COLOR_BGR2GRAY)

    output = cv2.inpaint(img, mask, 5, cv2.INPAINT_TELEA)

    return output
```

Slika 4.12. Funkcija *deleteSelectedObjects()* [12]

Pikseli izvan kutija su postavljeni kao crni, a unutar kutija su bijeli, što je vidljivo na slici 4.13.

```
61      mask = numpy.zeros_like(img)
62      for i in selectedObjects:
63          box = boxes[i - 1]
64          x, y, w, h = box[0], box[1], box[2], box[3]
65
66          |
67          mask[y:y + h + 1, x: x + w + 1] = 255
68
```

Slika 4.13. Postavljanje piksela [12]

Na 4.14. se vidi pretvaranje maske u 8-bitnu jedno-kanalnu sliku.

```
70      mask = cv2.cvtColor(mask, cv2.COLOR_BGR2GRAY)
71
```

Slika 4.14. Pretvaranje maske [12]

Završni dio koda koji vraća finalnu sliku s obrisanim objektom se nalazi na slici 4.15.

```
74      output = cv2.inpaint(img, mask, 5, cv2.INPAINT_TELEA)
75
76      return output
```

Slika 4.15. Kraj funkcije [12]

4.2. Main

„*Main.py*“ je glavni dio aplikacije.

Prvo su učitane sve potrebne biblioteke, te su napravljeni svi atributi i pozvane funkcije iz „*DetectAndRemove*“, što se vidi na slici 4.16.

```
1   from xmlrpc.client import boolean
2   from DetectAndRemove import *
3   import PySimpleGUI as pg
4
5   selectedObjects = []
6   img = []
7   boxes = []
8   numberOfObjects = 0
9   toggleDown = False
10
11  classes = getClasses()
12  net = getConfiguration()
13
```

Slika 4.16. Početak *„Maina“* [12]

Postavljanje teme pomoću funkcije *theme()* kao na slici 4.17.

```
15  pg.theme('DarkAmber')
```

Slika 4.17. Tema [12]

Kreiranje *layouta* aplikacije. Dodavanje teksta pomoću funkcije *text()* i tipki koje će se koristiti kada se aplikacija pokrene pomoću funkcije *Button()*, vidljivi su na slici 4.18.

```
layout = [
    [pg.Text('Pick an image'), pg.Input(key = '-IN-'), pg.FileBrowse(file_types = (('images', '*.png'),))
    , pg.Button('Load and detect')],
    [pg.Text('Select ids      '), pg.InputText(), pg.Button('Toggle visibility', disabled = True)],
]

```

Slika 4.18. *Layout* [12]

Kreiranje prozora aplikacije pomoću funkcije *Window()* koja kao jedan od argumenata prima *layout* je prikazano na slici 4.19.

```
window = pg.Window('Detect and Delete', layout)
```

Slika. 4.19. *Window* [12]

Petlja događaja koja se sastoji od više važnih dijelova koda nalazi se na slici 4.20.

```
while True:
    event, values = window.read()

    if event == 'Load and detect':
        if(values['-IN-']):
            toggleDown = False
            window['Toggle visibility'].update(disabled = False)
            displayImg, img, boxes, numberOfObjects = detectObjects(values['-IN-'], net, classes)
            cv2.imshow('Object detection', displayImg)

    elif event == 'Toggle visibility':

        if toggleDown == False:
            toggleDown = True

            listOfIds = values[0].strip(',').replace(" ", "").split(',')
            listOfIds = [s for s in listOfIds if s.isdigit()]
            listOfIds = [int(i) for i in listOfIds if int(i) > 0 and int(i) <= numberOfObjects + 1]


            outputImage = deleteSelectedObjects(listOfIds, boxes, img)
            cv2.imshow('Object detection', outputImage)


        elif toggleDown == True:
            toggleDown = False
            cv2.imshow('Object detection', displayImg)

    elif event == pg.WIN_CLOSED:
        break

```

Slika 4.20. *While* petlja [12]

S prvim uvjetom provjerava se je li kliknut „*Load and detect*“, a s drugim uvjetom se provjerava je li odabrana slika. To je vidljivo na slici 4.21.

```
    if event == 'Load and detect':

        if(values['-IN-']):
            toggleDown = False
            window['Toggle visibility'].update(disabled = False)
            displayImg, img, boxes, numberOfObjects = detectObjects(values['-IN-'], net, classes)
            cv2.imshow('Object detection', displayImg)

```

Slika 4.21. Uvjeti [12]

Na slici 4.22. provjerava se je li kliknut „*Toggle visibility*“.

```
41      elif event == 'Toggle visibility':
42
43          if toggleDown == False:
44              toggleDown = True
45
46              listOfIds = values[0].strip(',').replace(" ", "").split(',')
47              listOfIds = [s for s in listOfIds if s.isdigit()]
48              listOfIds = [int(i) for i in listOfIds if int(i) > 0 and int(i) <= numberOfObjects + 1]
49
50              outputImage = deleteSelectedObjects(listOfIds, boxes, img)
51              cv2.imshow('Object detection', outputImage)
52
53          elif toggleDown == True:
54              toggleDown = False
55              cv2.imshow('Object detection', displayImg)
56
57      elif event == pg.WIN_CLOSED:
58          break
```

Slika 4.22. Provjera [12]

Ovaj dio koda je napravljen kao osiguranje da se ne može unijeti ništa osim brojeva koji su veći od 0, a da su pritom manji ili jednaki najvećem ID-u. To je riješeno pomoću *fora* i *ifa.* Taj kod dostupan je na slici 4.23.

```
46              listOfIds = values[0].strip(',').replace(" ", "").split(',')
47              listOfIds = [s for s in listOfIds if s.isdigit()]
48              listOfIds = [int(i) for i in listOfIds if int(i) > 0 and int(i) <= numberOfObjects + 1]
49
```

Slika 4.23. Osiguravanje unosa [12]

Pozivanje detekcije objekta s funkcijom *imshow()* vidi se na slici 4.24.

```
50              outputImage = deleteSelectedObjects(listOfIds, boxes, img)
51              cv2.imshow('Object detection', outputImage)
52
```

Slika. 4.24. Detekcija [12]

Ako je „*toggleDown*“ istinit onda se pokazuje slika sa detektiranim objektima, te se on zatim postavlja na lažan. Slika se pokazuje sa uklonjenim objektima, što je prikazano na slici 4.25.

```
53          elif toggleDown == True:
54              toggleDown = False
55              cv2.imshow('Object detection', displayImg)
```

Slika 4.25. Dio koda [12]

Na slici 4.26. provjera se je li kliknut gumb „Exit“.

```
65          elif event == pg.WIN_CLOSED:
66              break
```

Slika 4.26. Dio koda [12]

Zatvaranje prozora, što ujedno predstavlja i završetak datoteke „*Main*“, prikazuje slika 4.27.

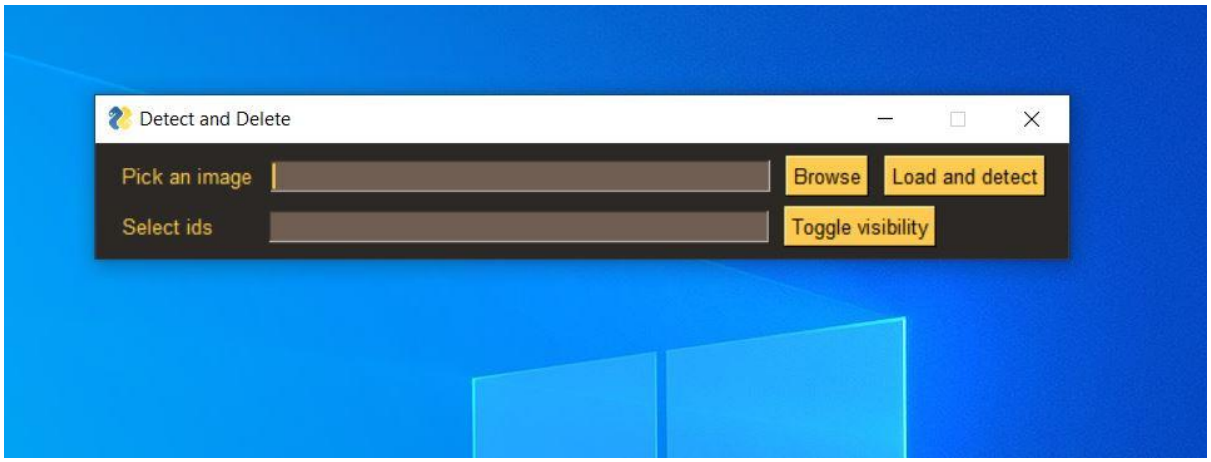
```
69      window.close()
```

Slika 4.27. Dio koda [12]

Kraj koda aplikacije.

5. KORIŠTENJE APLIKACIJE

Aplikacija se koristi tako da korisnik nakon pokretanja aplikacije prvo izabere sliku s koje želi ukloniti objekte. Nakon što je slika odabrana, programa označava objekte na slici, te ju vraća s obilježenim i numeriranim objektima. Potom korisnik odabir jedan ili više objekata koje želi ukloniti, te predaje njihove ID-eve. Aplikacija uklanja odabrane objekte i vraća sliku s uklonjenim objektima koju korisnik može dalje koristiti. Na slici 5.1. se vidi izgled aplikacije, te gumbovi za odabir, učitavanje i detekciju, te uklanjanje objekata.

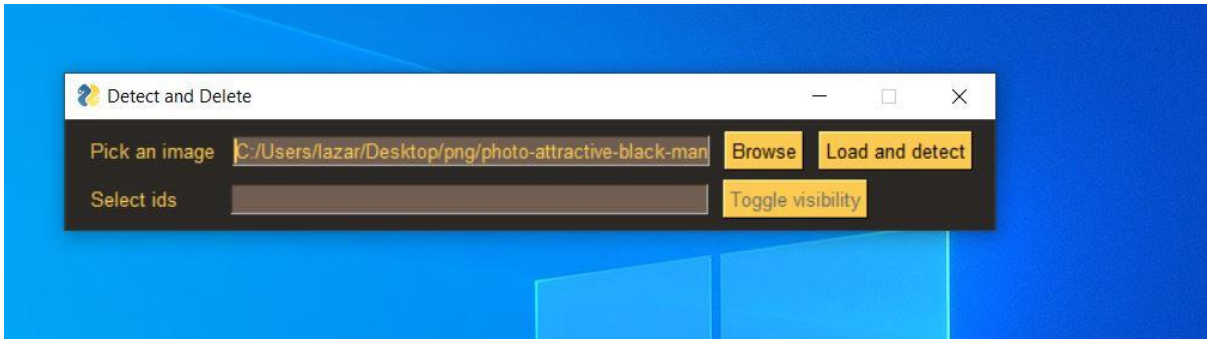


Slika 5.1. Aplikacija [12]

Slika 5.2. prikazuje sliku koja je uzeta kao prvi primjer za testiranje aplikacije.

Slika 5.2. Prvi primjer [9]

Na slici 5.3. se vidi kako je slika spremna za učitavanje, te detekciju objekata.

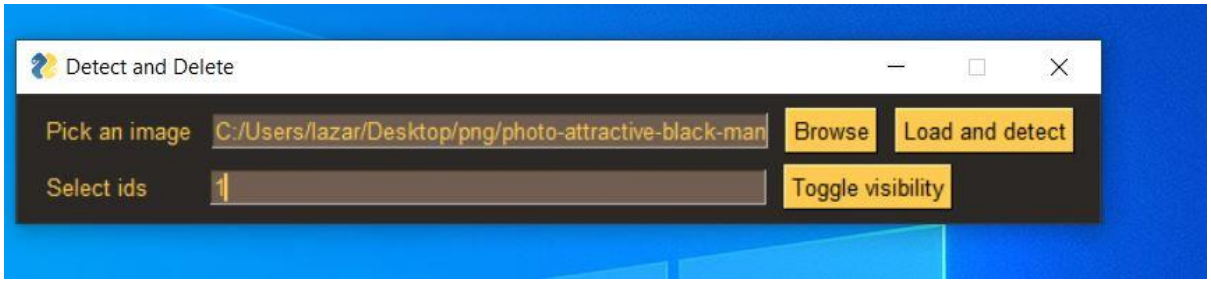


Slika 5.3. Odabrana slika [12]

Klikom na gumb *„Browse“* odabire se slika na kojoj se žele detektirati objekti. Nakon toga klikom na *„Load and Detect“* slika se učitava, te se na njoj označavaju objekti. Na slici 5.4. se vidi kako je aplikacija prepoznala objekt unutar slike.

Slika 5.4. Fotografija nakon detekcije objekta [12]

Upisuje se *ID* objekta, te klikom na gumb *„Toggle visibility“* on se uklanja sa slike. Kako se to radi prikazano je na slici 5.5.



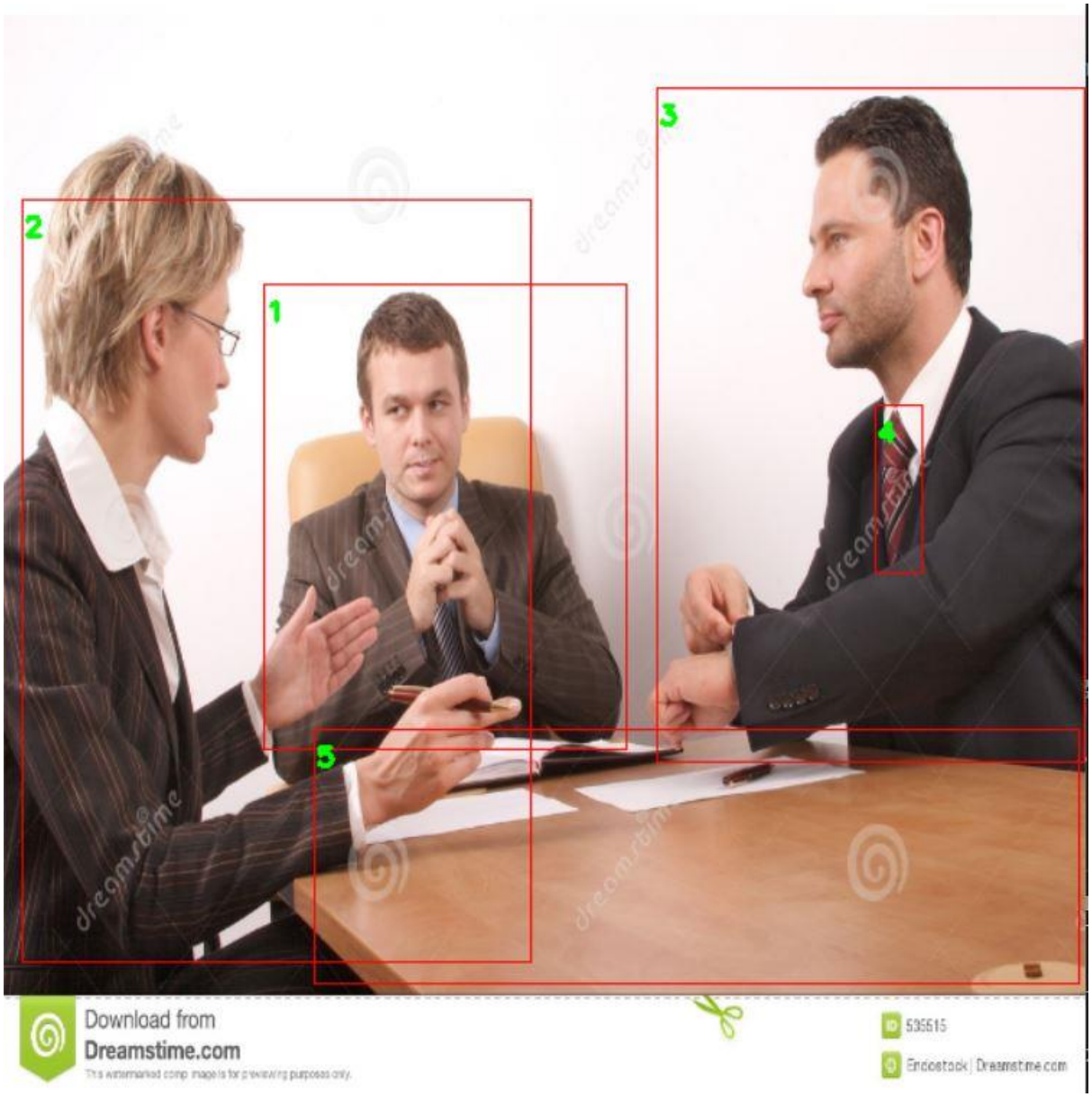
Slika 5.5. Odabir [12]

Konačan izgled slike s uklonjenim objektom se može jasno vidjeti na slici 5.6.

Slika 5.6. Finalna fotografija [12]

Za drugi primjer odabrana je slika 5.7. Kako izgleda slika s detektiranim objektima vidi se na 5.8. Ovim primjerom se želi pokazati da aplikacija može prepoznati i druge objekte osim ljudi. Slika na kojoj je obrisan objekt je slika 5.9. Na slici je obrisana kravata.

Slika 5.7. Originalna fotografija [10]



Slika 5.8. Fotografija s obilježenim objektima [12]

Slika 5.9. Finalna fotografija [12]

Na slici 5.10. se vidi rad aplikacije kada se objekt koji se želi ukloniti nalazi između dvije različite pozadine. Aplikacija ga je uspješno uspjela detektirati što se vidi na slici 5.11., te ga obrisati što se vidi na slici 5.12.

Slika 5.10. Čovjek na dvobojnoj pozadini [11]

Slika 5.11. Detekcija objekta [12]

Slika 5.12. Finalna fotografija [12]

6. ZAKLJUČAK

Ljudi danas u velikoj mjeri prate društvene mreže, te ako je njihova aktivnost na istima veća, žele da i njihove fotografije izgledaju što je savršenije moguće, te je zbog toga sve veća potreba, ali i dostupnost ovakvih aplikacija. Korisnici žele da svaka fotografija izgleda profesionalno, te su ih spremni uređivati koliko god je to potrebno.

Za izradu ove aplikacije korišten je programski jezik *Python*. Izrada aplikacije se odvila u *Microsoft Visual Studio Codu*. Kroz rad jasno je pokazan način stvaranja aplikacije korak po korak, na kraju je prikazan i sam rad aplikacije te su stavljeni rezultati. Aplikacije je ispitana za određene fotografije kako bi se pokazala točnost u radu.

Ova aplikacija se može koristiti za privatne svrhe. Nije poželjno da se koristi ako je fotografiju potrebno urediti do najsitnijih piksela. Aplikaciju je moguće dodatno nadograditi, tako da se doda veća preciznost prilikom detekcije, te uklanjanja objekta. Moguće je i dodavanje većeg izbora objekata za prepoznavanje.

7. SAŽETAK

Python je aplikacija za uklanjanje objekata iz fotografije. Aplikacija omogućuje korisniku odabir fotografije, te uklanjanje neželjenih objekata iz iste. Kao programski jezik, korišten je isključivo Python. Glavni cilj aplikacije je da se ista može koristiti i za jednostavne i za složenije fotografije. Aplikacija je rađena tako da je prvo napravljena datoteka za detekciju i brisanje objekata. Ta datoteka je koristila neke druge datoteke unutar sebe. Zatim je napravljena glavna datoteka, u kojoj je napravljen sam izgled aplikacije, te je ona pozivala datoteku za detekciju i brisanje. U analizi ispravnosti aplikacije koristile su se određene fotografije, koje su pokazale kako aplikacija ispravno i točno radi.

Ključne riječi: aplikacija, fotografija, objekt, *OpenCV, Python*

8. ABSTRACT

Removing objects from the photo

Python application for removing objects from photos. The application allows the user to select a photo and remove unwanted objects. As programming language, only Python is used. The main goal of the application is that it can be used for both simple and more complex pohotos. The application was created in such way that first is created file for detection and removing objects. That file used some other files inside it. Then the main file was created, in which the appearance of the application itself was created, and it used the file for detection and removing. In the analysis of the correctness of the application, certain photos were used, which showed how application works correctly and accurately.

Keywords: application, object, OpenCV, photo, Python

9. LITERATURA

[1] ADVA Soft GmbH, TouchRetouch: Photos Retouch, Apple, SAD, 2019., https://apps.apple.com/us/app/touchretouch/id373311252 [15.9.2022.]

[2] Adobe, Adobe Photoshop Fix, Google, SAD, 2021., https://play.google.com/store/apps/details?id=com.adobe.adobephotoshopfix&hl=en_US&gl=US [15.9.2022.]

[3] Snapseed, The Best Photo Editing App – Snapseed App, Snapseed, SAD, 2011., https://snapseed.online/ [16.9.2022.]

[4] PIXLR, Uređivač fotografija, animacija i dizajn, pixlr, SAD, 2021., https://pixlr.com/hr/ [16.9.2022]

[5] CyberLinks, Capture. Edit. Retouch., CyberLink, SAD, 2011., https://www.cyberlink.com/products/photodirector-photo-editing-software-365/features_en_US.html [16.9.2022]

[6] A. C. Stross-Radschinski, PythonBrochure, Plone & Python, SAD, 2014., https://brochure.getpython.info/media/releases/prerelases/pythonbrochure-20140309_21-51-45rz92-proofreading-dl/view [15.9.2022.]

[7] Python, What is Python? Executive Summary, python, SAD, 2020., https://www.python.org/doc/essays/blurb/ [15.9.2022.]

[8] OpenCV, OpenCV Face Recognition, OpenCV, SAD, 2018., https://opencv.org/ [15.9.2022.]

[9] VS Code, Code editing. Redefined., Microsoft, SAD, 2020., https://code.visualstudio.com/ [15.9.2022.]

[10] Rajnis09, Python OpenCV, geeksforgeeks, SAD, 2023., https://www.geeksforgeeks.org/python-opencv-cv2-rectangle-method/ [15.9.2022.]

10. IZVOR SLIKA

[1] https://www.podfeet.com/blog/2018/09/touchretrouch-ios/

[2] https://play.google.com/store/apps/details?id=com.adobe.adobephotoshopfix&hl=hr&gl=US

[3] https://play.google.com/store/apps/details?id=com.niksoftware.snapseed&hl=hr&gl=US

[4] https://lmhmod.com/en/pixlr-mod

[5] https://www.downloadcrew.com/download/35264/cyberlink_photodirector_essentials

[6] https://en.wikipedia.org/wiki/Python_%28programming_language%29

[7] https://en.wikipedia.org/wiki/OpenCV

[8] https://code.visualstudio.com/docs/getstarted/tips-and-tricks

[9] https://img.freepik.com/premium-photo/photo-attractive-black-man-with-piercing-with-eye-patches-smiles-naked-torso-isolated-brown-color-background_341052-710.jpg?w=2000

[10] https://thumbs.dreamstime.com/z/business-meeting-3-persons-535515.jpg

[11] https://media.gettyimages.com/photos/mature-afro-man-portrait-on-colorful-background-picture-id1143707416?s=612x612

[12] Izradio autor

11. PRILOZI

Prilog 1. Završni rad u datoteci docx

Prilog 2. Završni radu u datoteci pdf

Prilog 3. Programski kod aplikacije

Prilog 4. Datoteka „coco-names“

Prilog 5. Datoteka „frozen_inference_graph.pb“

Prilog 6. Datoteka „ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt“