

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
SVEUČILIŠTA U ZAGREBU

INTERAKTIVNA RAČUNALNA GRAFIKA

Dokumentacija laboratorijskih vježbi

Marko Golub

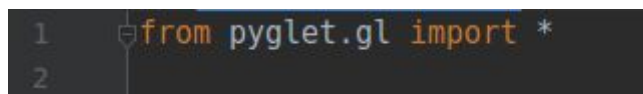
Zagreb, svibanj 2020.

1. Korištene tehnologije i alati:

Svi programi pisani su koristeći programski jezik Python verzije 3.6.7. i na operacijskom sustavu Ubuntu 16.04. te pokretani u razvojnoj okolini PyCharm verzije 2019.2.4.

1.1. PyOpenGL

PyOpenGL je platforma koja povezuje programski jezik Python i OpenGL. Povezivanje je ostvareno preko *pyglet.gl* paketa koji je izdan pod BSD licencom. Uključivanje paketa obavlja se pozivom naredbe



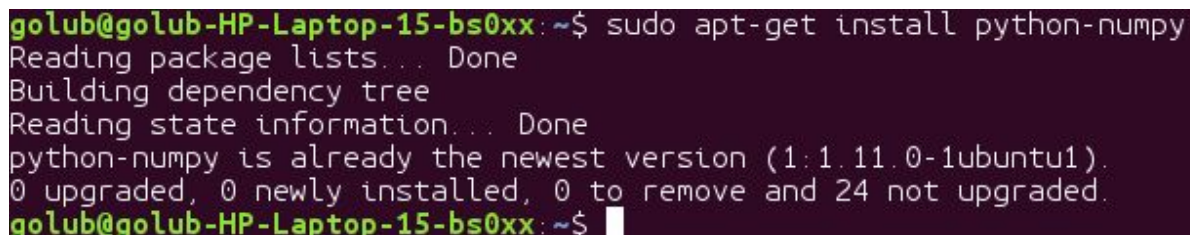
```
1 from pyglet.gl import *
2
```

Slika 1. Uključivanje biblioteke

*“from pyglet.gl import *”*.

1.2. NumPy

NumPy je projekt otvorenog koda nastao 2005. godine s ciljem razvoja programske podrške koja omogućavanja numeričko računanje pomoću programskog jezika Python. Projekt je izdan pod BSD licencom, razvijen je otvoreno i smješten u javno GitHub skladište¹. Za inačicu operacijskog sustava Ubuntu 16.04 instalaciju je moguće obaviti pozivom naredbe *“sudo apt-get install python-numpy”* izravno u ljusci,



```
golub@golub-HP-Laptop-15-bs0xx:~$ sudo apt-get install python-numpy
Reading package lists... Done
Building dependency tree
Reading state information... Done
python-numpy is already the newest version (1:1.11.0-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 24 not upgraded.
golub@golub-HP-Laptop-15-bs0xx:~$
```

Slika 2. Instalacija paketa

¹ <https://github.com/numpy/numpy>

no postoje i drugi načini² ovisno o operacijskom sustavu. Uvoz paketa u radnu okolinu obavlja se naredbom `import numpy`. Kroz rad paket će biti uvežen naredbom `import numpy as np` radi preglednosti koda.

2. Vježbe

2.1. Osnovne operacije u RG³

2.1.1. Osnovno o vježbi

U računalnoj grafici površine objekata prikazujemo aproksimacijom točaka i poligona. Prikladan način za prikaz takvih podataka su vektori i matrice. Cilj je ove vježbe upoznati korisnika s osnovnim operacijama nad vektorima i matricama kroz prva dva zadatka i baricentričnim koordinatama u trećem zadatku.

2.1.2. Zadatak 1

Zadatak sadrži primjere: zbrajanja vektora, skalarnog produkta vektora, vektorskog produkt vektora, operacije normiranja vektora, izračun vektora suprotnog smjera, zbrajanje matrica, množenje matrica, izračun transponirane matrice te izračun inverzne matrice.

Pokretanjem izvođenja programa ispisuju se rezultati operacija nad, u uputama zadanim, vektorima i matricama.

U daljnjem radu moguće je program napisati bez korištenja *numpy* biblioteke.

² <https://scipy.org/install.html>

³ RG - računalna grafika

2.1.3. Zadatak 2

Program ispisuje rješenje sustava tri jednažbe s tri nepoznanice u obliku $[x \ y \ z] = [a \ b \ c]$, gdje su nepoznanice x, y, z , a parametri a, b, c realni brojevi.

Izvođenje je obavljeno pomoću numpy biblioteke pretvorbom sustava u matrice.

Nedostatak programa je manjak parsiranja ulaza. U slučaju unosa pogrešnih argumenata, program baca iznimku.

```
Original exception was:
Traceback (most recent call last):
  File "/media/golub/5ee0dd7e-0856-4e9f-82f8-59ca97092b2f/home/golub/Documents/FEF/sem6/IRG/lab/lab1/zad2.py", line 6, in <module>
    f = [int(x) for x in first]
  File "/media/golub/5ee0dd7e-0856-4e9f-82f8-59ca97092b2f/home/golub/Documents/FEF/sem6/IRG/lab/lab1/zad2.py", line 6, in <listcomp>
    f = [int(x) for x in first]
ValueError: invalid literal for int() with base 10: 'a'
```

Slika 3. Ispis na zaslonu pogreške u unosu podataka

U budućem radu moguće je program ograditi takvih slučajeva hvatanjem iznimaka i navođenjem korisnika točnom unosu podataka dodatnim ispisima na zaslonu.

2.1.4. Zadatak 3

Pokretanjem izvršavanja koda od korisnika zahtjeva unos 3 vrha trokuta A, B, C te dodatne točke T. S obzirom na unesene točke, program na zaslonu ispisuje baricentrične koordinate.

```
1, 1, 1
3, 3, 1
2, 10, 1
2, 2, 1
[t1 t2 t2] = [0.5 0.5 0. ]

Process finished with exit code 0
```

Slika 4. Primjer izvođenja traženja baricentričnih koordinata

Nedostatak programa je loša reakcija na pogrešan unos podataka - bacanje iznimke. U budućem radu moguće je program ograditi takvih slučajeva hvatanjem iznimaka, ali i omogućiti rad s realnim brojevima.

2.2. Pravac, linija

2.2.1. Osnovno o vježbi

Crtanje je linija jedna od najčešćih operacija u RG stoga je važno tu operaciju obavljati točno. Vježba ilustrira korištenje jednog od algoritama za crtanje linija te upoznaje sa sintaksom: linije crta pomoću `“pyglet.gl.GL_POINTS”`, a pomoćne linije pomoću `“pyglet.gl.GL_LINES”`.

2.2.2. Zadatak 1

Pokretanjem izvršavanja koda na zaslonu korisnika otvara se prazan prozor. Pritiskom na lijevi gumb miša zadaje se početna točka, a ponovnim pritiskom na lijevi gumb zadaje se završna točka linije koju program crta.

U usporedbi s uputom, algoritam je dovršen te je program točno crta linije pod svim kutevima. To je omogućeno kroz dvije funkcije ovisno o položaju točaka: `“draw_line_X”` i `“draw_line_Y”`.

```

# Algorithm that draws good lines from -45° to +45°.
# Loop iterates by the x variable.
def draw_line_X(xS, yS, xE, yE):...

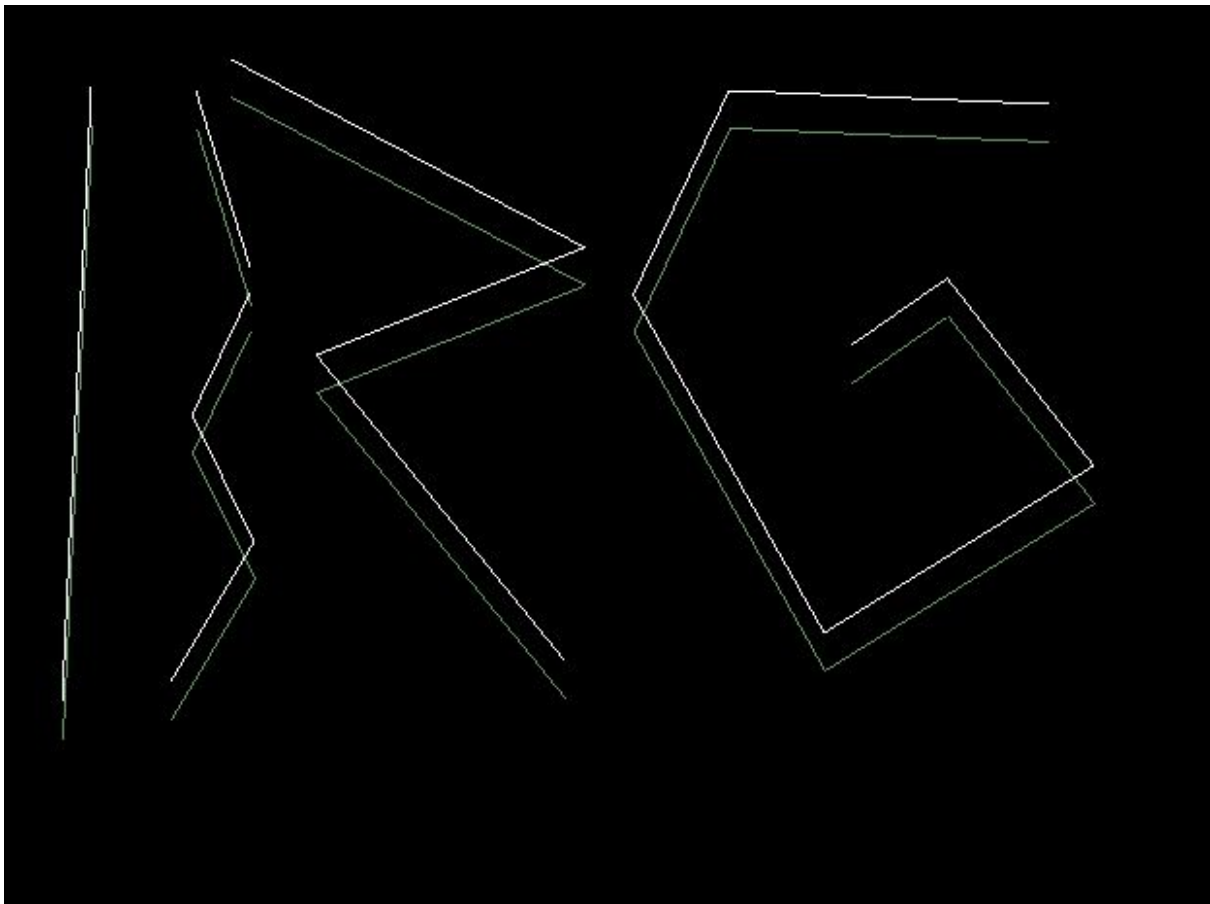
# Algorithm that draws good lines from +45° to -45°.
# Loop iterates by the y variable.
def draw_line_Y(xS, yS, xE, yE):...

# Algorithm that calls draw_line_X or draw_line_Y based on points' coordinates.
def draw_line(xS, yS, xE, yE):...

# Method that draws following line raised by 20 pixels.
def draw_following_line(xS, yS, xE, yE):
    glBegin(GL_LINES)
    glVertex2i(xS, yS+20)
    glVertex2i(xE, yE+20)
    glEnd()

```

Slika 4. Funkcije `draw_line_X` i `draw_line_Y`



Slika 5. Primjer izvođenja koda i crtanja pojedinih linija

2.3. Konveksan poligon

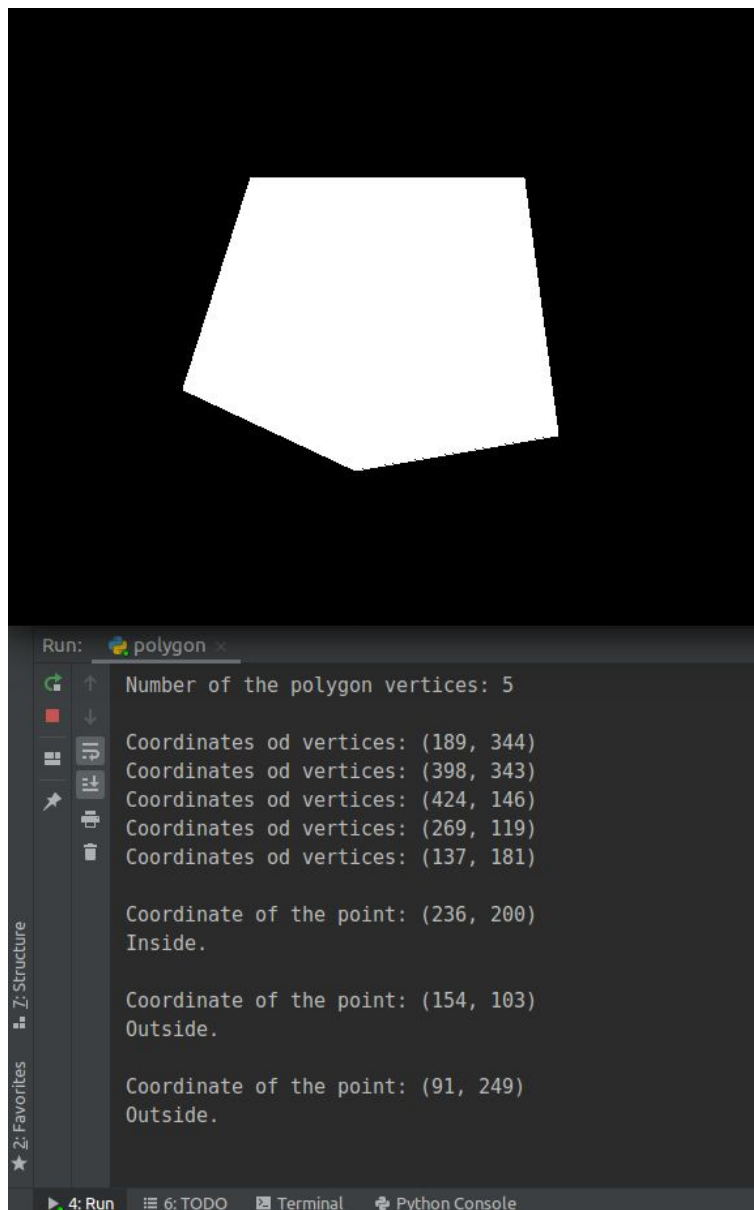
2.3.1. Osnovno o vježbi

Vježba ilustrira crtanje poligona, računanje odnosa točke i samog poligona i korištenje jednog od algoritama za bojanje poligona.

2.3.2. Zadatak 1

Pokretanjem izvršavanja koda na zaslonu korisnika otvara se prazan prozor te se od korisnika očekuje unos točaka, pritiskom na lijevi gumb miša, u smjeru kazaljke na satu. Pritiskom na desni gumb miša postupak zadavanja točaka završava te je korisniku omogućeno zadatavi točke u prozoru pritiskom na lijevi gumb miša, a program ispisuje odnos zadanih točaka i poligona na zaslon.

Nekoliko je nedostataka programa: ispravno radi samo ako za konveksne poligone i točke zadane u smjeru kazaljke na satu. U budućem radu nad programom moguće je program generalizirati kako bi ispravno bojao i konveksne i konkavne likove neovisno o smjeru zadavanja točaka. Također, potrebno je uzeti u obzir crtanje i bojanje vodoravnih linija.



Slika 6. Primjer izvođenja koda i bojanja konkavnog poligona i odnosa točke s obzirom na sam poligon

2.4. Ravnina tijela

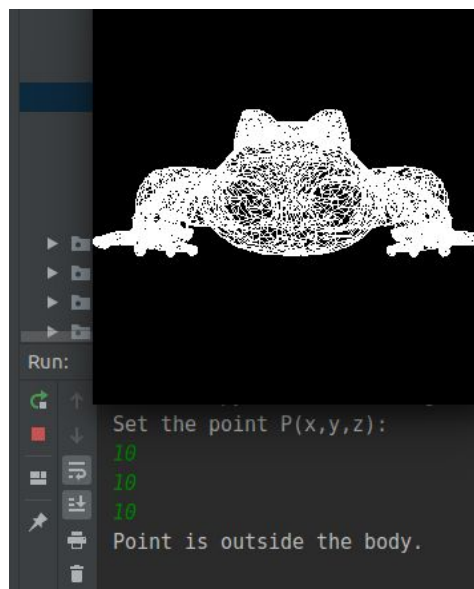
2.4.1. Osnovno o vježbi

Ravninu određuju tri nekolinearne točke. Samim time, potrebne su nam bar tri točke kako bismo zadali ravninu tijela. U ovoj vježbi korišten je dio .obj zapisa kako bi se zadali trodimenzionalni objekti. Vježba izgrađuje topološku strukturu tijela i ispituje odnos zadane točke i tijela.

2.4.2. Zadatak 1

Pokretanjem izvršavanja koda učitava se unaprijed zadani objekt iz .obj datoteke: redak čije je prvo slovo "v" označava vrhove, redak čije je prvo slovo "f" označava poligone, a "#" komentare. Od korisnika se traži unos koordinata trodimenzionalne točke te program na zaslon ispisuje odnos točke i objekta.

Nedostatak programa je nemogućnost korisnika da učitava različite objekte. Moguće rješenje je omogućavanje unosa naziva datoteke i njeno otvaranje. Također, potrebno je ograditi se od pogrešnog unosa.



Slika 7. Primjer izvođenja koda za objekt "frog.obj"

2.5. Transformacija pogleda i perspektivna projekcija

2.5.1. Osnovno o vježbi

Vježba osigurava mogućnost promatranja scene iz bilo koje pozicije kamere.

2.5.2. Zadatak 1

Prikazani su preslikavanje točaka sustava scene u sustav oka i projekcija perspektive pomoću dvije funkcije:

“*calculate_view_transformation_matrix*” i

“*calculate_view_perspective_projection_matrix*”. Također pomoću funkcije “*on_key_press*” omogućena je promjena koordinata točke očišta ovisno o pritisnutom gumbu.

```
@window.event
def on_key_press(key, modifiers):
    global O, G, window, shift

    if(key == pygame.key.UP):
        O[1] = O[1] + shift
        window.clear()
        draw_object(O, G)
    elif(key == pygame.key.DOWN):
        O[1] = O[1] - shift
        window.clear()
        draw_object(O, G)
    elif(key == pygame.key.LEFT):
        O[0] = O[0] - shift
        window.clear()
        draw_object(O, G)
    elif(key == pygame.key.RIGHT):
        O[0] = O[0] + shift
        window.clear()
        draw_object(O, G)

@window.event
def calculate_view_transformation_matrix(O, G):...

@window.event
def calculate_perspective_projection_matrix(O, G):...
```

Slika 8. Opisane funkcije i prikaz gumba za koje se aktiviraju transformacije

Pokretanjem izvršavanja koda na zaslonu se otvara prazan prozor. Prva transformacija i crtanje objekta događa se pritiskom na bilo koji od 4 gumba strelica.

Upravo to je i najveći nedostatak programa - nakon pokretanja, bez sudjelovanja korisnika, ne događa se baš ništa. Moguće rješenje je iscrtavanje objekta istog trena ili ispis uputa korištenja na zaslon.

2.6. Krivulja Bezijera

2.6.1. Osnovno o vježbi

U programu su Bezierove krivulje definirane polinomima Bernsteina te je prikazana je animacija tijela po krivulji.

2.6.2. Zadatak 1

Pokretanjem izvršavanja koda na zaslonu korisnika otvaraju se dva prozora: prvi prozor iscrtava Bezierovu krivulju, a drugi je prozor prazan. Animacija objekta po krivulji ostvaruje se pritiskom na gumb "Enter". Funkcija "C" računa binomne koeficijente te je ograđena kako bi se poštivala matematička pravila.



```
def C(n, k):  
    if k+1 > n or k < 0: raise ArithmeticError
```

Slika 9. Iznimka unutar funkcije C

Nedostatke ovog programa moguće je riješiti s nekoliko poboljšanja. Za početak, pomoglo bi iscrtavanje kontrolnih linija u prozoru Bezierove krivulje. Dokle god bi se ostalo na pristupu zadavanja točaka bez interakcije s korisnikom, moguće poboljšanje je odrediti krivulju koja tvori zatvoren poligon kako bi se dobilo kontinuirana i tečna animacija objekta.

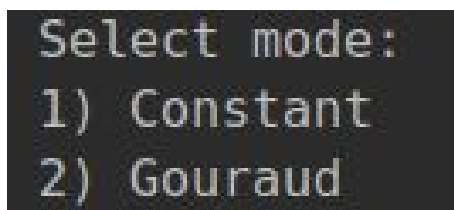
2.7. Sjenčanje tijela

2.7.1. Osnovno o vježbi

Prikazan je algoritam određivanja stražnjih poligona kako bi se ubrzalo izvođenje postupka crtanja i algoritmi konstantnog i Gouraud-ovo sjenčanja objekata.

2.7.2. Zadatak 1

Pokretanjem izvršavanja koda korisniku se daje opcija izbora načina izvođenja programa.



Slika 10. Ispis izbora načina izvođenja programa

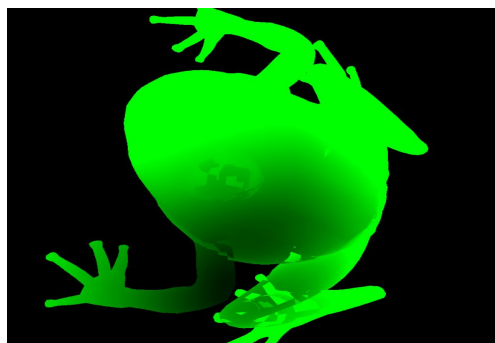
```
def main():
    global mode
    global Q, G, I
    print("Select mode:\n1) Constant\n2) Gouraud")
    while mode not in [1, 2]: mode = int(input())
    draw_object(0, G)
    pygame.app.run()
```

Slika 11. Izbor načina izvođenja programa

Od korisnika će se tražiti unos podatka sve dok on ne bude ispravan. Za odabir konstantnog sjenčanja prikazuje se tijelo takvo da sva tri vrha poligona imaju jednak intenzitet prethodno izračunan za svaki vrh. Brzina izvođenja prihvatljiva je za oba načina rada programa, no moguća ne dodatna optimizacija.



Slika 11. Prikaz u prvom načinu rada



Slika 12. Prikaz u drugom načinu rada

Moguća poboljšanja su usavršavanje brisanja stražnjih poligona.

2.8. Fraktali

2.8.1. Osnovno o vježbi

Program nudi jednostavan algoritam za računanje točaka i prikaz dva fraktalna skupa - Mandelbrotov i Julijev.

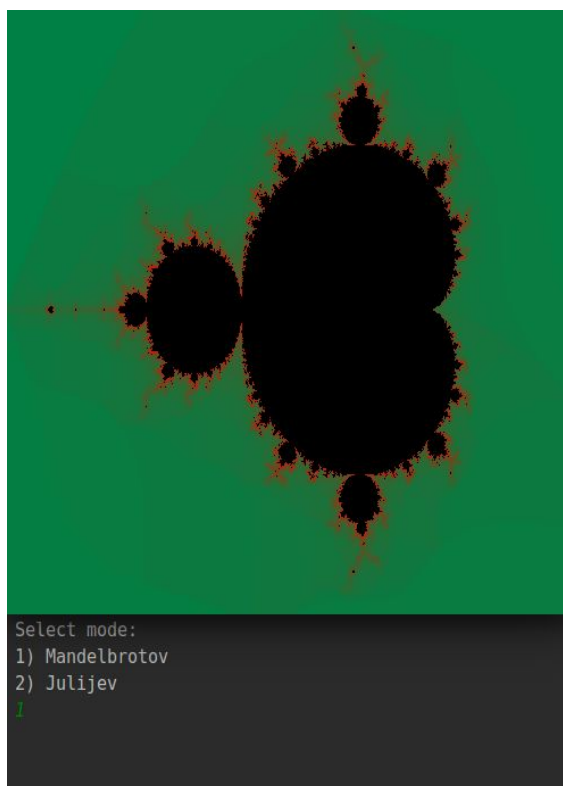
2.8.2. Zadatak 1

Pokretanjem programa na zaslonu korisnika prikazuje se prazan prozor i na zaslonu se ispisiuje opcija odabira dva načina rada te se očekuje korisnički unos odabira načina izvođenja: “1” za Mandelbrotov skup točaka, “2” za Julijev skup točaka.

Algoritmi⁴ su objedinjeni u jedan algoritam te se ovisno o odabranom načinu rada pobuđuje odgovarajući dio koda.

Brzina izvođenja je prihvatljiva, no moguće poboljšanje programa bilo bi omogućavanje korisničkog odabira pragova i razlučivosti zaslona.

⁴ <http://www.zemris.fer.hr/predmeti/irg/knjiga.pdf>



Slika 13. Prikaz izvođenja prvog načina rada



Slika 14. Prikaz izvođenja drugog načina rada