SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1402

RAZVOJ PROGRAMSKIH SUSTAVA U OKRUŽENJU KHOROS PRO 2001

Josip Haluška

Zagreb, veljača 2003.

Sadržaj

1. Uvod

Proces obrade podataka ima važnu ulogu kod raspoznavanja uzoraka, inteligentnih sustava, morfologije, sažimanja podataka, prikaza slika u medicini, dobivanja vizualnih riješenja znanstvenih poblema, telekomunikacijama, robotici, biologiji i disciplinama koje se bave pročavanjem reljefa ili okoliša.

Pad cijena računala i digitalnih sustava, uz povećanje performansi, omogućio je širenje procesa obrade podataka na navedena područja.

Kod vizualizacije podataka, koriste se svi raspoloživi resursi (računala, mreže, korisnička sučelja, razni algoritmi) i kombiniraju sa ljudskim opažanjem, kako bi se riješio problem. Vizualizacija omogućuje potpuno razumijevanje i uvid u podatke koji se proučavaju. Kako bi se olakšao taj proces, mora postojati okolina za brzu izradu prototipova (eng. fast prototyping), potrebni su operatori za obradu podataka i vizualno predočavanje rezultata.

Khoros je nastao kao istraživački projekt na University of New Mexico. Prva verzija programa pojavila se sredinom 1991. godine. Popularnost je stekao nudeći velik broj složenih algoritmama, koji su bili namjenjeni prvenstveno obradi slike i obradi signala.

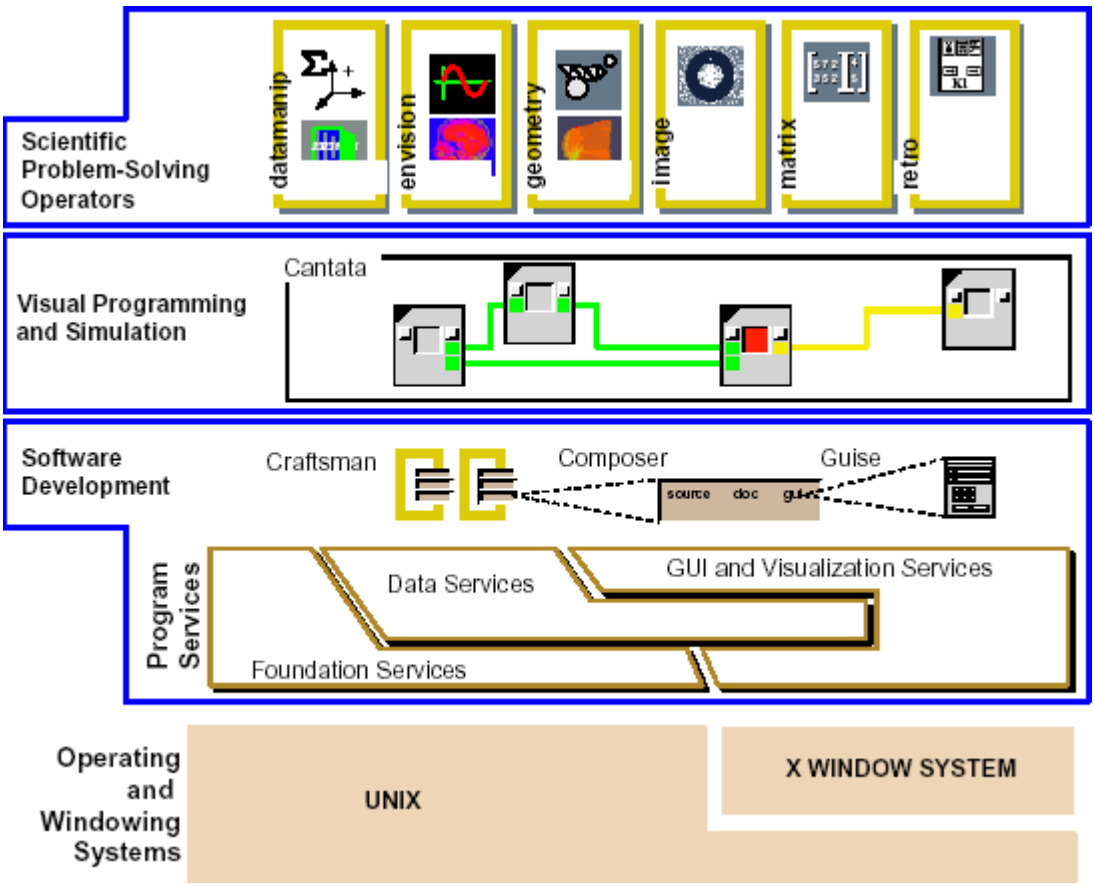
Zadnja verzija Khoros sustava je Khoros Pro 2001 v2, pomoću koje su napravljeni svi primjeri prikazani u diplomskoj radnji. Radnja je podijeljena na dva osnovna dijela. U prvom dijelu se kroz promjere opisuje korištenje Khorosa (2. poglavlje) i Cantate (3. poglavlje), a u drugom dijelu primjena Cantate za riješavanje problema izdvajanja registarske tablice automobila iz slike (4. poglavlje).

2. Khoros

Khoros je integrirani programski paket odnosno razvojna okolina, koja omogućuje obradu podataka i njihovo proučavanje. Osnovne značajke:

- sadrži velik broj gotovih rutina (bilblioteka) koje korisnik može upotrijebiti za rješavanje specifičnih problema;
- grafičko sučelje omogućuje brzi razvoj prototipova;
- sustav je proširiv i prenosiv na različite platforme;
- uključena je podrška na različitim razinama složenosti, prilagođena običnom korisniku, pa sve do programerskog eksperta;
- postoji velik broj korisnika, koji međusobno komuniciraju i razmijenjuju svoje znanje i gotova rješenja.

Različiti korisnički nivoi prikazani su na slici 2.1.



Slika 2.1. Različite razine korištenja paketa Khoros.

Na najnižoj razina Khorosa nalaze se Toolbox alati. Toolbox predstavlja kolekciju programa i bilblioteka koja čini entitet. Obično se Toolbox određuje prema primjeni, tako da se programi za obradu slike mogu nalaziti u jednom, a programi za obradu signala u drugom Toolboxu. Craftsman Composer i Guise su Khoros alati namjenjeni za rad sa Toolboxima.

Na srednjoj razini nalazi se Cantata, alat za vizualno programiranje i simuliranje. Programi se sastoje od objekata nazvanih Glyph. Program kojeg Glyph predstavlja naziva se operator. Jednostavnim odabirom i povezivanjem Glyphova korisnik formira mrežu, koja čini grafički program.

Na najvišoj razini nalaze se gotovi opertori za riješavanje problema iz različitih znanstvenih područja.

Mogućnost hijerarhije, iteracije, upravljanja tokom i promjene parametara koji se pojavljuju u izrazima čine Cantatu jakim alatom za simulaciju i brzu izradu prototipova. Ključ jednostavnosti rješavanja vizualnih problema je u tome što se ne koristi nekakav alat oslonjen na tekstualni prikaz, nego baš grafičko sučelje.

Osim glyphova koji služe za upravljanje tokom podataka, postoje i vizualne procedure radi mogućnosti hijerarhije.

Varijable korisnik može mijenjati direktno, ili se one mijenjaju kroz matematičke izraze za vrijeme izvođenja programa.

Za korisnike koji razvijaju aplikacije, postoji potpuna podrška razvojne okoline. Krajnji proizvod ima sve mogućnosti Khorosa, od grafičkog sučelja pa do pristupa velikom broju datoteka različith formata.

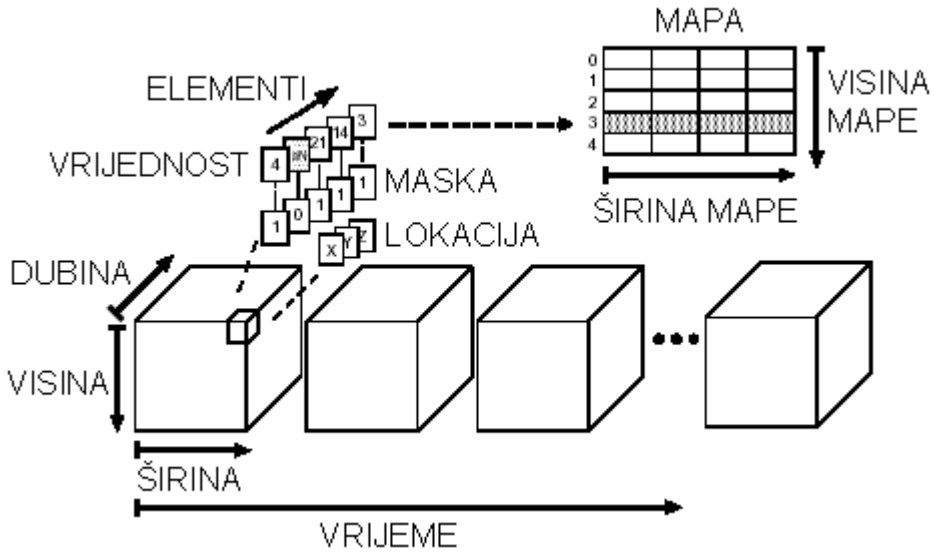
Višeslojna arhitektura Khorosa i koncept servisnih programa omogućuje jednostavno rješavanje složenih problema, pri čemu pojedinosti i poslovi na razini operacijskog sustava ostaju skriveni od krajnjeg korisnika.

2.1. Polimorfni model podataka

Khoros ima svoju posebnu strukturu podataka, namjenjenu za rad sa višedimenzionalnim podacima, koja se naziva polimorfni model podataka (eng. Polymorphic Data Model). Pomoću nje moguće je opisati jednodimenzionalne (1D) podatke, odnosno signale kao što je zvuk, dvodimenzionalne (2D) podatke kao što su slike, ili trodimenzionalne (3D) podatke koji sadrže volumensku informaciju.

2.1.1. Opis modela

Polimorfni model podataka se može gledati i kao vremenski niz volumena u prostoru koji se sastoje od pet različitih podatkovnih segmenata. Podatkovni segmenti su: vrijednost, lokacija, vrijeme, maska i mapa.



Slika 2.2. Polimorfni model podataka.

Svaki segment ima posebnu namjenu, ali svi ne moraju biti navedeni. Objekt podataka može sadržavati neku kombinaciju segmenata i da još uvijek pripada polimorfnom modelu.

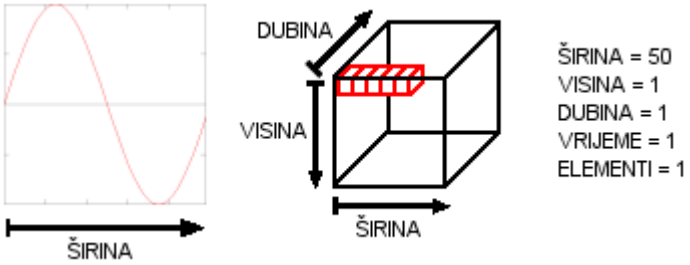
U segmentu vrijednosti svaki element ima implicitnu poziciju određenu vektorom (visina, širina, dubina, vrijeme, elementi). Umjesto vrijednosti elemanata, može postojati pokazivač na neku poziciju segmenta mape na kojoj se nalazi vrijednost.

Elementi segmenta maske označavaju da li je vrijednost na istoj poziciji unutar segmenta vrijednosti, maskirana ili ne.

Segment lokacije određuje poziciju vektora vrijednosti u prostoru.

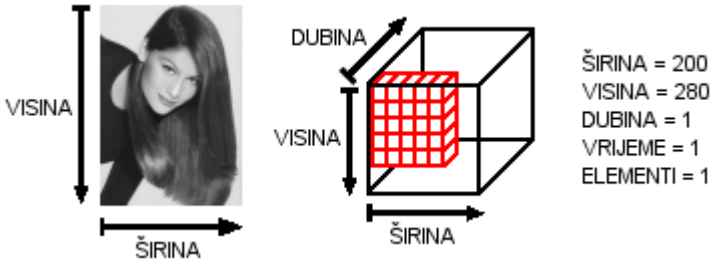
2.1.2. Pohranjivanje različitih objekata u polimorfni model

Signali mogu zauzeti bilo koju dimenziju polimorfnog modela, ali se obično pohranjuju kroz širinu segmenta vrijednosnosti.

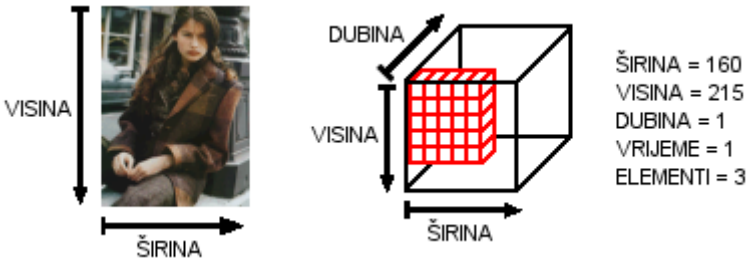


Slika 2.3. Prikaz signala u polimorfnom modelu.

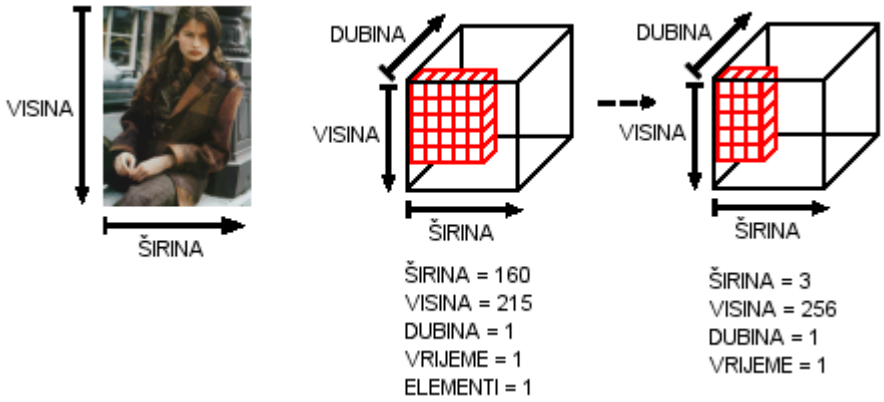
Slike se pohranjuju preko širine i visine segmenta vrijednosti. Svaki element slike može biti indeks na segment mape ili može činiti vektor preko dimenzije elemenata.



Slika 2.4. Prikaz sive slike u polimorfnom modelu.

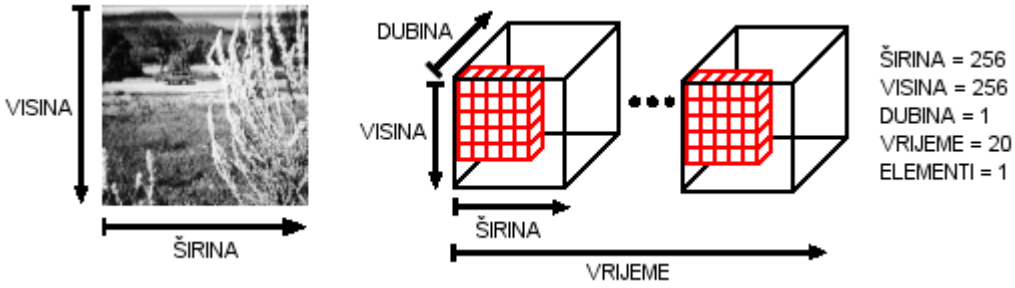


Slika 2.5. Prikaz RGB slike u polimorfnom modelu.



Slika 2.6. Prikaz RGB slike pomoću mape boja u polimorfnom modelu.

Animacija se sastoji od niza slika pohranjenih preko dimenzije vremena.



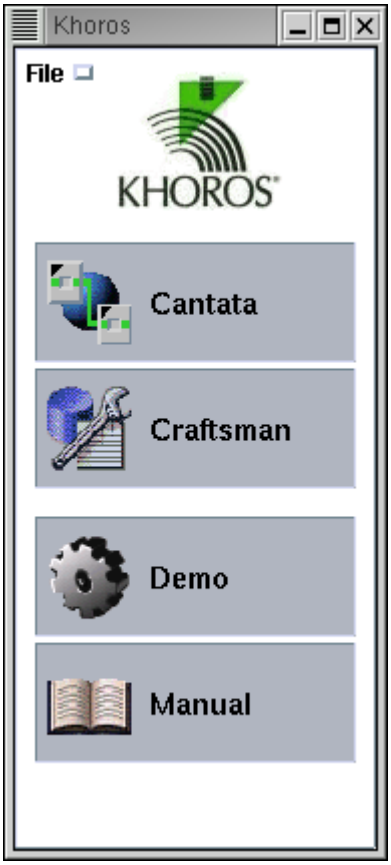
Slika 2.7. Prikaz animacije pomoću 25 sivih slika u polimorfnom modelu.

2.2. Pokretanje Khoros sustava

Khoros se pokreće upisom naredbe khoros, pod komandnom linijom Unix sustava.

```
%khoros &
```

"%" označava prompt Unix sustava, a "&" pokreće program u pozadini, tako da komandna linija ostaje slobodna za nove naredbe. Nakon pokretanja na ekranu se pojavljuje osnovni prozor Khorosa.



Slika 2.8. Glavni prozor Khorosa.

Cantata je grafička radna okolina, a Craftsman služi za održavanje i razvoj unutar sustava, posebno za upravljanje operatorima. Cantata i Craftsman se mogu pokrenuti preko glavnog prozora Khoros sustava, ali i preko komandne linije kao samostalne aplikacije.

Odabirom dviju zadnjih opcije prozora, dobivaju se demonstracije primjera koji dolaze sa Khoros paketom, ili uputstva koja se nalaze na web stranicama proizvođača.

Operatori Khoros sustava mogu se izvršavati preko tri različita sučelja:

- CLUI (Command Line User Interface), komandna linija;
- GUI (Graphical User Interface), grafičko sučelje;
- Cantata, okolina za vizualno programiranje.

2.3. Rad u komandnoj liniji

Jedan od načina rada je rad pod komandnom linijom, odnosno CLUI. To je najniža razina rada sa Khorosom, gdje se naredbe unose preko komandne linije Unix sustava, zajedno sa svim parametrima.

2.3.1. Prikaz slike sa sivim tonovima

Slika "laetitia1.bmp" je pomoću rutine kformats pretvorena u KDF (Khoros 2.0 Data Format) format.

```
%kformats –i laetitia1.bmp –o laetitia1.kdf –kdf
```

Kod poziva rutina iz komandne linije argument "-i" određuje ulaznu, a argument "-o" izlaznu datoteku. Kod rutine kformats argument "-kdf" označava ciljni format slike.

Za ovu i ostale rutine objašnjenje se dobije navođenjem argumenta "-U" za osnovne i "-usage" za detaljnije upute.

```
%kformats –U

(Brief usage; use [-usage] for standardized Khoros arguments)

Usage for kformats:
"Conversion Between the Formats Supported in Khoros 2"

% kformats
  -i       (infile)  Input data object
  -o       (outfile) Resulting output data object

Required Mutually Exclusive Group -
Specify exactly one of the 14 options:
  [-jpg]   (flag)    Output as JPG
  [-kdf]   (flag)    Output as KDF (Khoros 2.0 Format)
  [-pnm]   (flag)    Output in Portable Any Map (PNM) format
  [-viff]  (flag)    Output as Viff (Khoros 1.0 Image Format)
  [-pcx]   (flag)    Output in PC Paintbrush (PCX) format
  [-xbm]   (flag)    Output in X Bitmap (XBM) format
  [-ascii] (flag)    Output in ASCII format
...
```

Rutina kfileinfo daje osnovne informacije koje su pohranjene u zaglavlju datoteke. Prikazano je zaglavlje dobivene slike "laetitia1.kdf".

```
%kfileinfo –i laetitia1.kdf

# Name: laetitia1.kdf
# Comment:
# End Comment
# Machine Architecture: Little Endian IEEE
# Date Created: Tue Dec 10 12:18:21  2002
# Storage Format: kdf
# Format Description: Khoros 2.0 Data Format (kdf)
# Sub-Object Position:   0 , 0 , 0 , 0 , 0
# World Coordinate Point Size:   1 , 1 , 1 , 1 , 1
#
# Color Space Model:  0 (invalid)
# Has Alpha Channel:   0
#
# maskedValuePresentation:  0
#
# -- Value Data --
# Data Type: Unsigned Byte
# Size: Width = 200, Height = 280, Depth = 1, Time = 1, Elements = 1
```

Iz zaglavlja se može vidjeti da je slika širine 200 i visine 280 točaka.

```
# Size: Width = 200, Height = 280 ...
```

Isto tako je vidljivo da svaki slikovni element može poprimiti jednu od 256 cjelobrojnih vrijednosti.

```
# Data Type: Unsigned Byte
```

Slika se može vidjeti pomoću putimage operatora.

```
%putimage –i laetitia1.kdf &
```

Slika 2.9. Slika prikazana operatorom putimage.

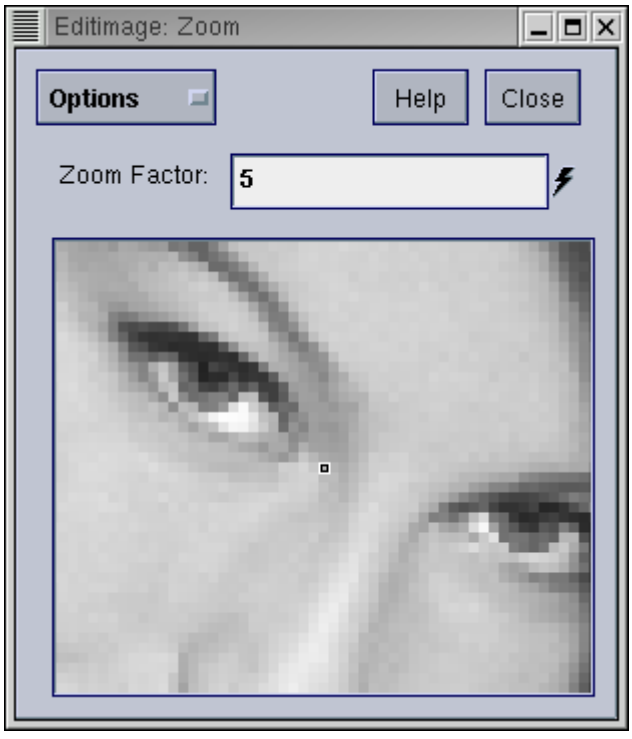
Kada se preko slike prelazi pokazivačem, pri dnu prozora se mijenjaju koordinate slikovnih elemenata i njihova vrijednost (širina x visina = vrijednost sive razine). Gornji lijevi kut odgovara koordinatama (0, 0), a donji desni (200, 280). Manje vrijednosti odgovaraju tamnijoj, a veće vrijednosti svijetlijoj sivoj razini. Putimage prozor se zatvara preko [File] [Quit].

Operator editimage omogućava potpuno istraživanje i mijenjanje objekta.

```
%editimage –i laetitia1.kdf &
```

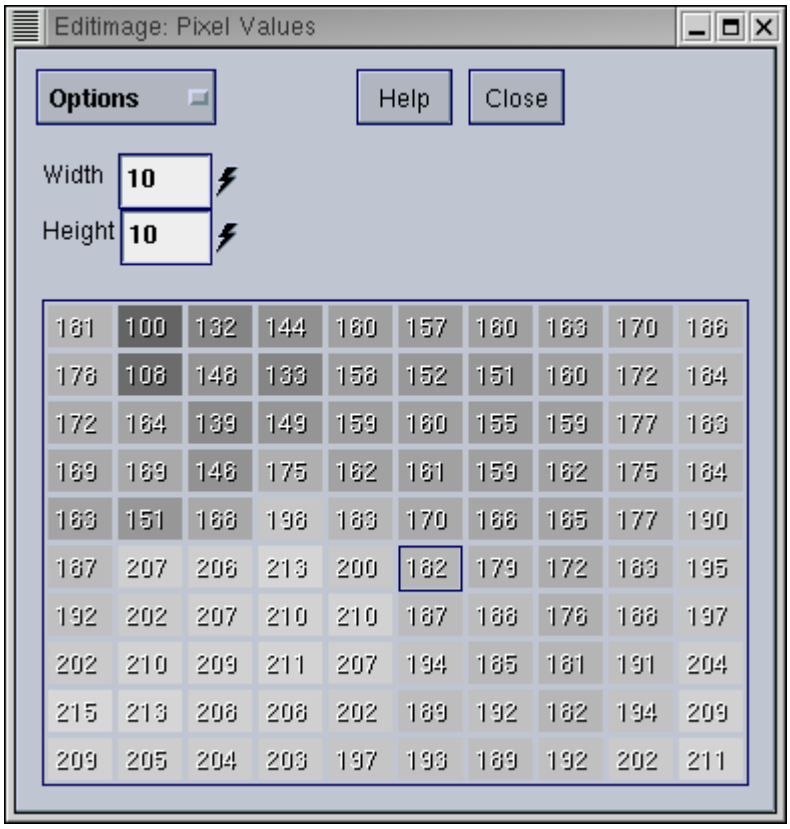
Slika 2.10. Slika prikazana operatorom editimage.

Prozor sa povećanom slikom dobije se izborom [View] [Zoom...].



Slika 2.11. Povećana slika.

Prozor sa vrijednostima sivih razina dobije se izborom [View] [Pixel Values...].



Slika 2.12. Prikazane su vrijednost slikovnih elementata.

Prelazeći pokazivačem preko slike orginalnog prozora mijenja se uvećana slika i mijenjaju se vrijednosti sivih razina, prateći poziciju pokazivača. Operator editimage se zatvara preko [File] [Exit].

2.3.2. Prikaz slike u boji

Slika "laetitia2.bmp" je pomoću rutine kformats pretvorena u KDF format.

```
%kformats –i laetitia2.bmp –o laetitia2.kdf –kdf
```

Prikazano je zaglavlje dobivene slike "laetitia2.kdf".

```
%kfileinfo –i laetitia2.kdf

# Name: laetitia2.kdf
# Comment:
# End Comment
# Machine Architecture: Little Endian IEEE
# Date Created: Tue Dec 10 12:20:22  2002
# Storage Format: kdf
# Format Description: Khoros 2.0 Data Format (kdf)
# Sub-Object Position:   0 , 0 , 0 , 0 , 0
# World Coordinate Point Size:   1 , 1 , 1 , 1 , 1
#
# Color Space Model:  0 (invalid)
# Has Alpha Channel:   0
#
# maskedValuePresentation:  0
#
# -- Value Data --
# Data Type: Unsigned Byte
# Size: Width = 160, Height = 215, Depth = 1, Time = 1, Elements = 1
#
# -- Map Data --
# Data Type: Unsigned Byte
# Size: Width = 3, Height = 256, Elements = 1,
#       Depth Dimension = 1, Time Dimension = 1
```

U opisu zaglavlja se vidi da je osim segmenta vrijednosti prisutan i segment mape boja. Vrijednosni segment može pokazivati na jednu od 256 boja koje su opisane mapom.

```
# -- Value Data --
# Data Type: Unsigned Byte
```

Mapa sadrži 256 boja koje su sastavljene od tri komponente.

```
# -- Map Data --
# Size: Width = 3, Height = 256 ...
```

Svaka komponenta boje može poprimiti jednu od 256 cjelobrojnih vrijednosti.

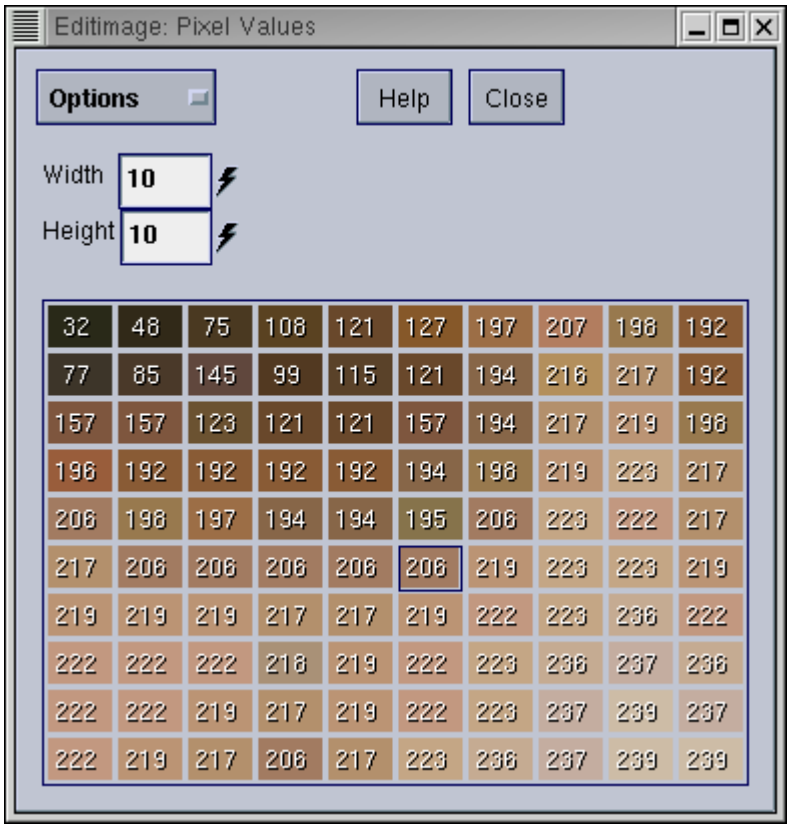
```
# -- Map Data --
# Data Type: Unsigned Byte
```

Nakon što se pokrene operator editimage, osim prije navedenih prozora potrebno je otvoriti još jedan preko [View] [Map Values...], kako bi se vidjele komponente boje.

```
%editimage –i laetitia2.kdf &
```

Slika 2.13. Slika prikazana operatorom editimage.

Slika 2.14. Povećana slika.



Slika 2.15. Prikazane su vrijednost slikovnih elemenata.

Slika 2.16. Prikazane su RGB komponente.

2.3.3. Prikaz animacije

Informacije o datoteci koja sadrži niz slika dobiju se pomoću rutine kfileinfo.

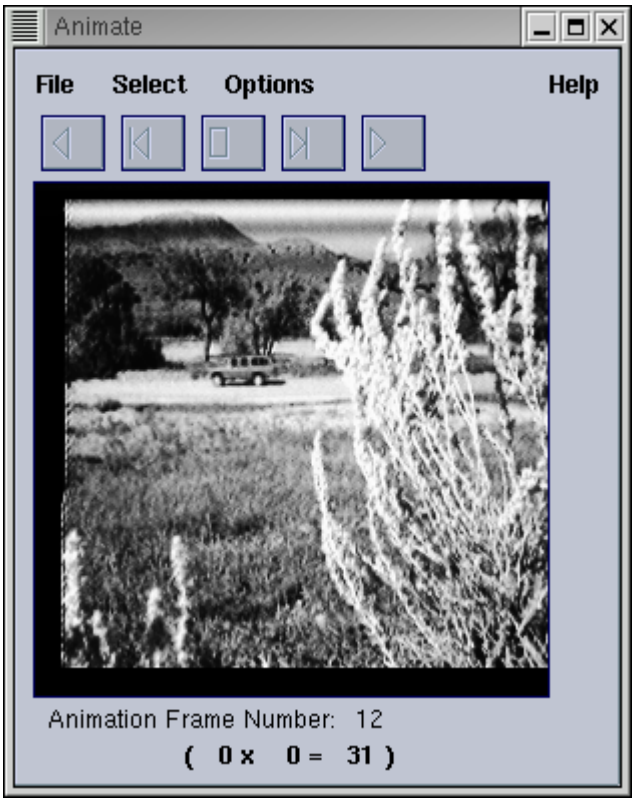
```
%kfileinfo –i sequences:bushes

# Name: /usr/local/kp2001v2/datamanip/datamanip/data/sequences/bushes.kdf
# Comment:
This is an image sequence provided by Sandia National Labs, taken by
Greg Donohoe and Ed Hoover in the Sandia Labs playground.
# End Comment
# Machine Architecture: Big Endian IEEE
# Date Created: Sat Aug 27 19:22:30 MDT 1994
# Storage Format: kdf
# Format Description: Khoros 2.0 Data Format (kdf)
# Sub-Object Position:   0 , 0 , 0 , 0 , 0
# World Coordinate Point Size:   1 , 1 , 1 , 1 , 1
#
# Color Space Model:  0 (invalid)
# Has Alpha Channel:   0
#
# fspare1:  0
# fspare2:  0
# ispare1:  0
# ispare2:  0
# maskedValuePresentation:  0
# xvimageMapEnable:  0
# xvimageMapScheme:  0
# xvimageSubrowSize:  0
#
# -- Value Data --
# Data Type: Unsigned Byte
# Size: Width = 256, Height = 256, Depth = 1, Time = 20, Elements = 1
```

Datoteka "bushes.kdf" dolazi kao primjer zajedno sa Khoros paketom i već se nalazi u KDF formatu tako da pretvorba nije potrebna. Iz zaglavlja se vidi da datoteka sadrži 20 slika visine i širine 256 točaka, od kojih svaka može poprimiti jednu od 256 cjelobrojnih vrijednosti.

```
# Data Type: Unsigned Byte
# Size: Width = 256, Height = 256 ... Time = 20
```

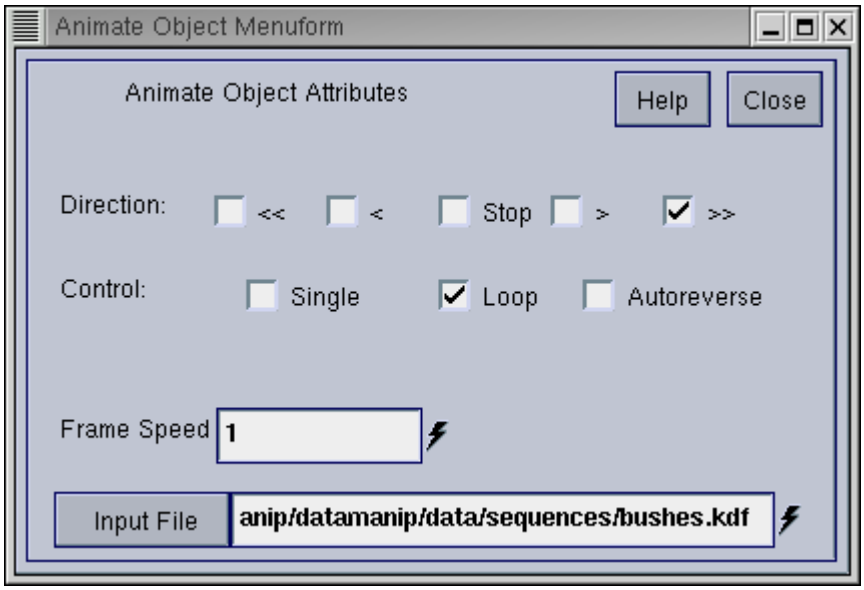
Operator animate omogućuje prikaz niza slika, odnosno animaciju.

```
%animate –i sequence:bushes &
```



Slika 2.17. Prikaz animacije pomoću operatora animate.

Koristeći gumbe iznad slike moguće je gledati animaciju prema natrag ili prema naprijed neprekidno, ili sliku po sliku. Preko izbornika [File] [Options...] može se mijenjati i brzina izmjene slika.



Slika 2.18. Podešavanje parametara operatora animate.

2.3.4. Pretvorba ASCII formata u KDF

Na primjeru će se pokazati pretvorba ASCII datoteke u KDF format. Pomoću tekstualnog editora potrebno je kreirati datoteku "lines.ascii" i pohraniti ju. Prikazan je sadržaj datoteke.

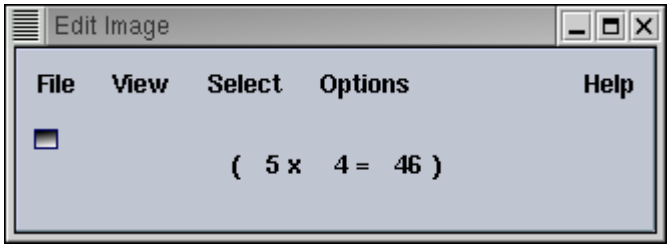
```
01 02 03 04 05 06 07 08 09 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
```

Pretvorba se napravi sa rutinom kformats kao u prethodnom primjeru.

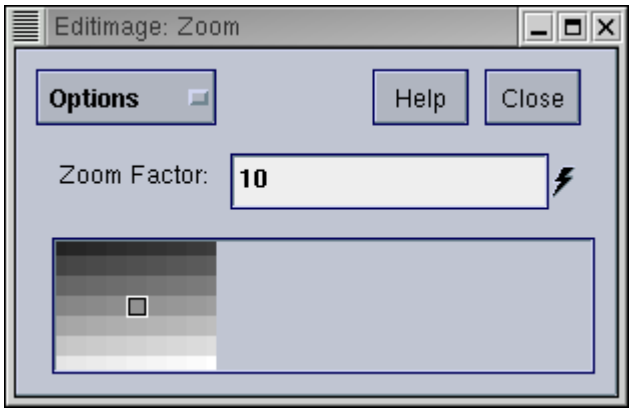
```
%kformats –i lines.ascii –o lines.kdf –kdf
```

Pomoću operatora editimage može se vidjeti slika te uvjeriti da vrijednosti sivih razina odgovaraju vrijednostima ulazne ASCII datoteke.

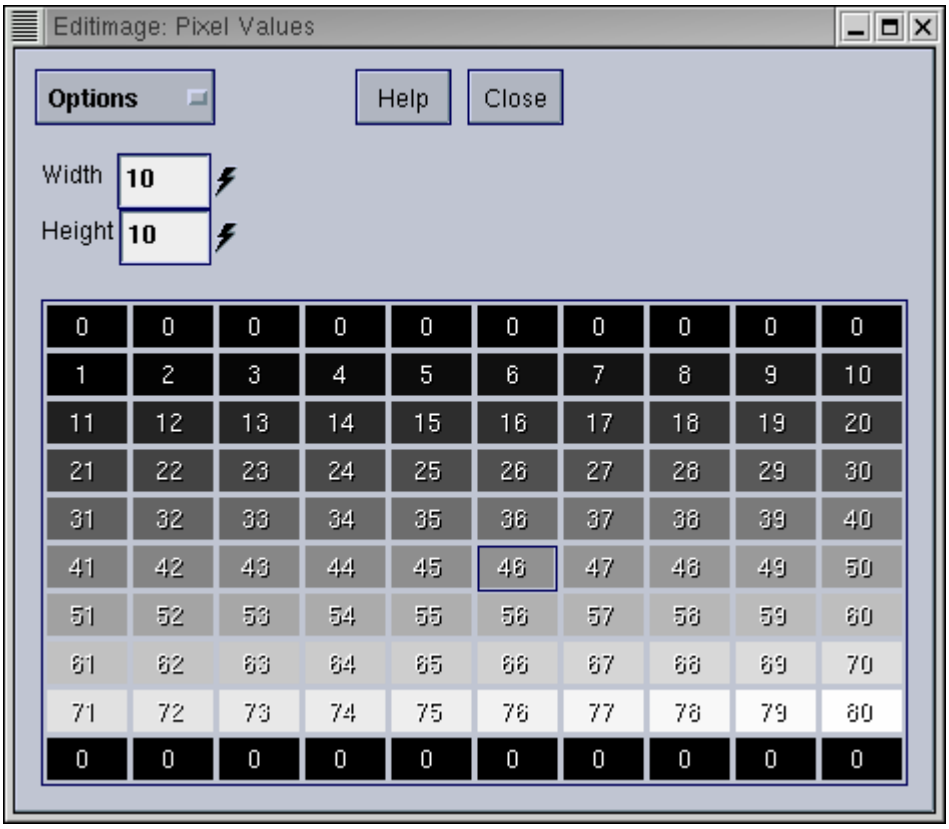
```
%editimage –i lines.kdf &
```



Slika 2.19. Slika prikazana operatorom editimage.



Slika 2.20. Povećana slika.



Slika 2.21. Prikazane su vrijednost slikovnih elemenata.

2.4. Korištenje grafičkog sučelja

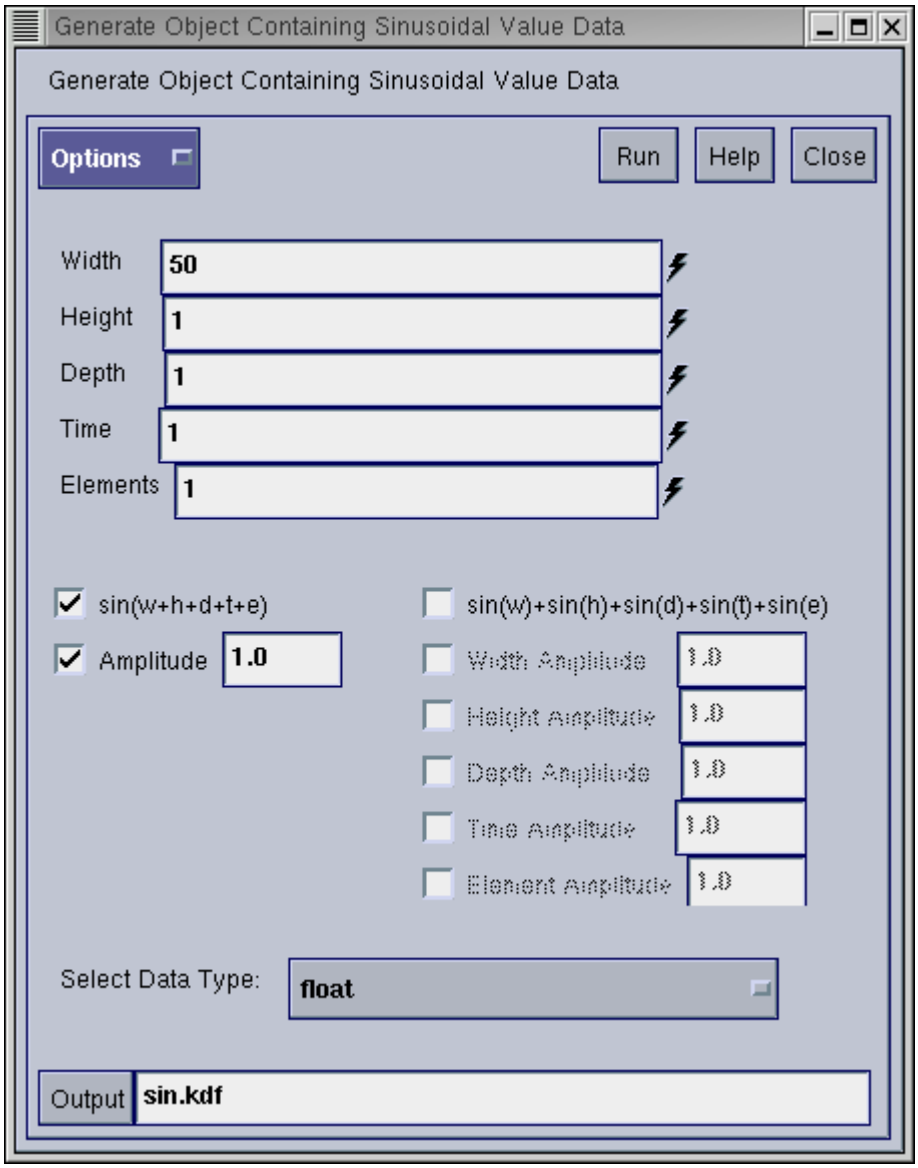
Za razliku od komandne linije, rad pod grafičkim sučeljem (GUI Graphical User Interface) je ugodniji i jednostavniji od načina rada pri komandnoj liniji (CLUI Command Line User Interface). Primjerima će se pokazati razlika između ta dva načina rada.

2.4.1. Generiranje signala i pretvorba u ASCII format

U primjeru koji slijedi generirat će se sinusoida pomoću kgsin operatora i pretvoriti u ASCII format. Operator bi se kao i što je do sada bio slučaj mogao potpuno izvesti iz komandne linije, ali zbog većeg broja argumenata, grafičko sučelje daje bolji i jednostavniji pregled.

```
%kgsin -gui &
```

Argument "-gui" označava da se operator pokreće u grafičkom sučelju.



Slika 2.22. Prozor s parametrima operatora kgsin.

Potrebno je postaviti parametare kao na slici 2.22, te upisati ime "sin.kdf" za izlaznu datoteku. Klikom na [Run] generira se i izvršava u komandnoj liniji ekvivalentan kgsin sa svim navedenim atributima.

```
%kgsin -wsize 50 -hsize 1 ... -o "sin.kdf" ...
```

Opis zaglavlja dobivene datoteke se može vidjeti sa kfileinfo.

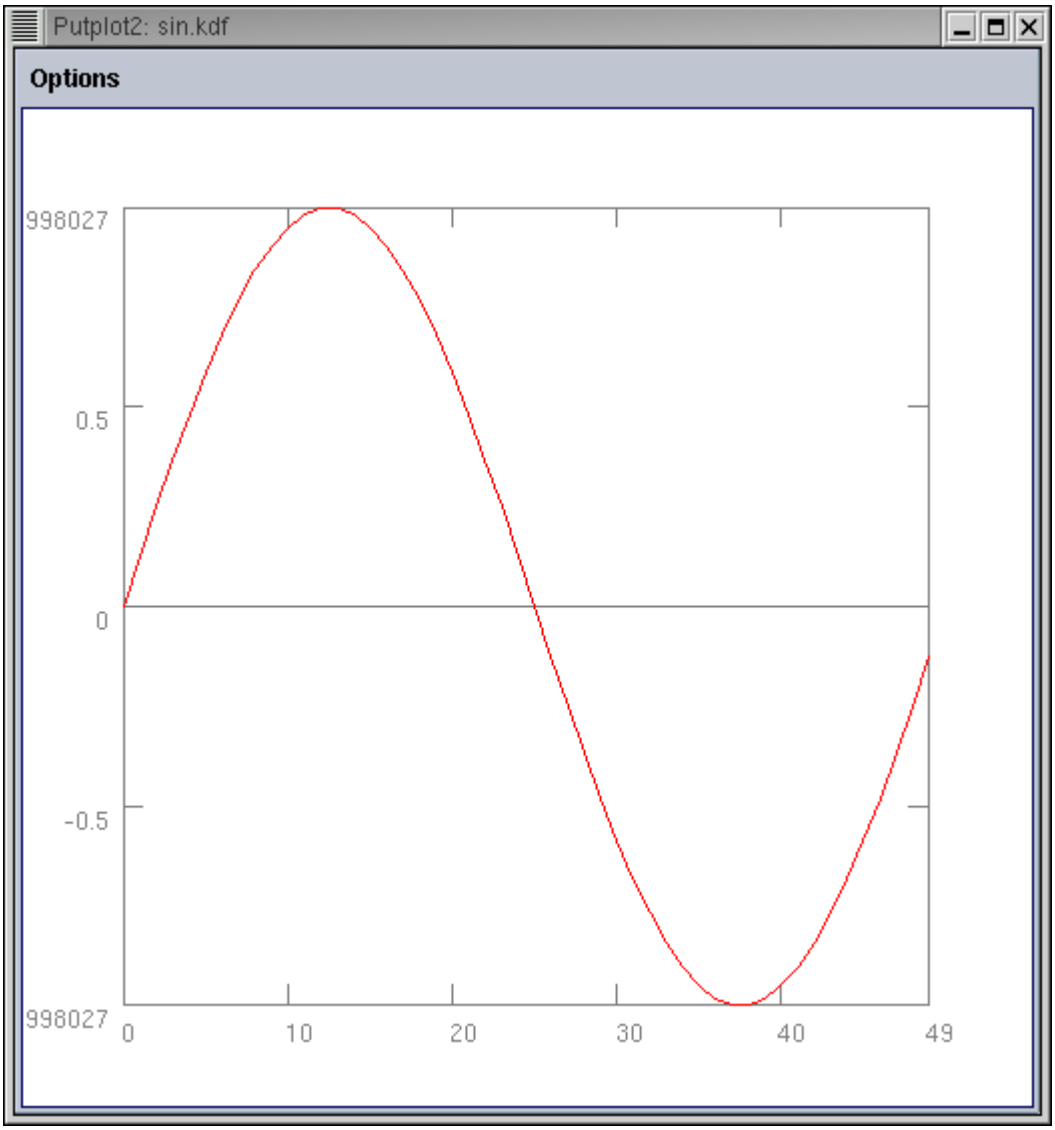
```
%kfileinfo –i sin.kdf

# Name: sin.kdf
# Comment:
# End Comment
# Machine Architecture: Little Endian IEEE
# Date Created: Wed Dec 11 13:06:54  2002
# Storage Format: kdf
# Format Description: Khoros 2.0 Data Format (kdf)
# Sub-Object Position:   0 , 0 , 0 , 0 , 0
# World Coordinate Point Size:   1 , 1 , 1 , 1 , 1
#
# Color Space Model:  0 (invalid)
# Has Alpha Channel:   0
#
# maskedValuePresentation:  0
#
# -- Value Data --
# Data Type: Float
# Size: Width = 50, Height = 1, Depth = 1, Time = 1, Elements = 1
```

Kao rezultat dobije se sinusoida sa 50 uzoraka realnih vrijednosti.

```
# Data Type: Float
# Size: Width = 50 ...
```

Pomoću operatora putplot2 može se vidjeti slika "sin.kdf".

```
%putplot2 –i sin.kdf &
```



Slika 2.23. Prozor operatora putplot2 sa nacrtanom sinusoidom.

Pretvorba slike sinusoide u ASCII format napravi se uporabom kformats.

```
%kformats -i sin.kdf -o sin.ascii -ascii
```

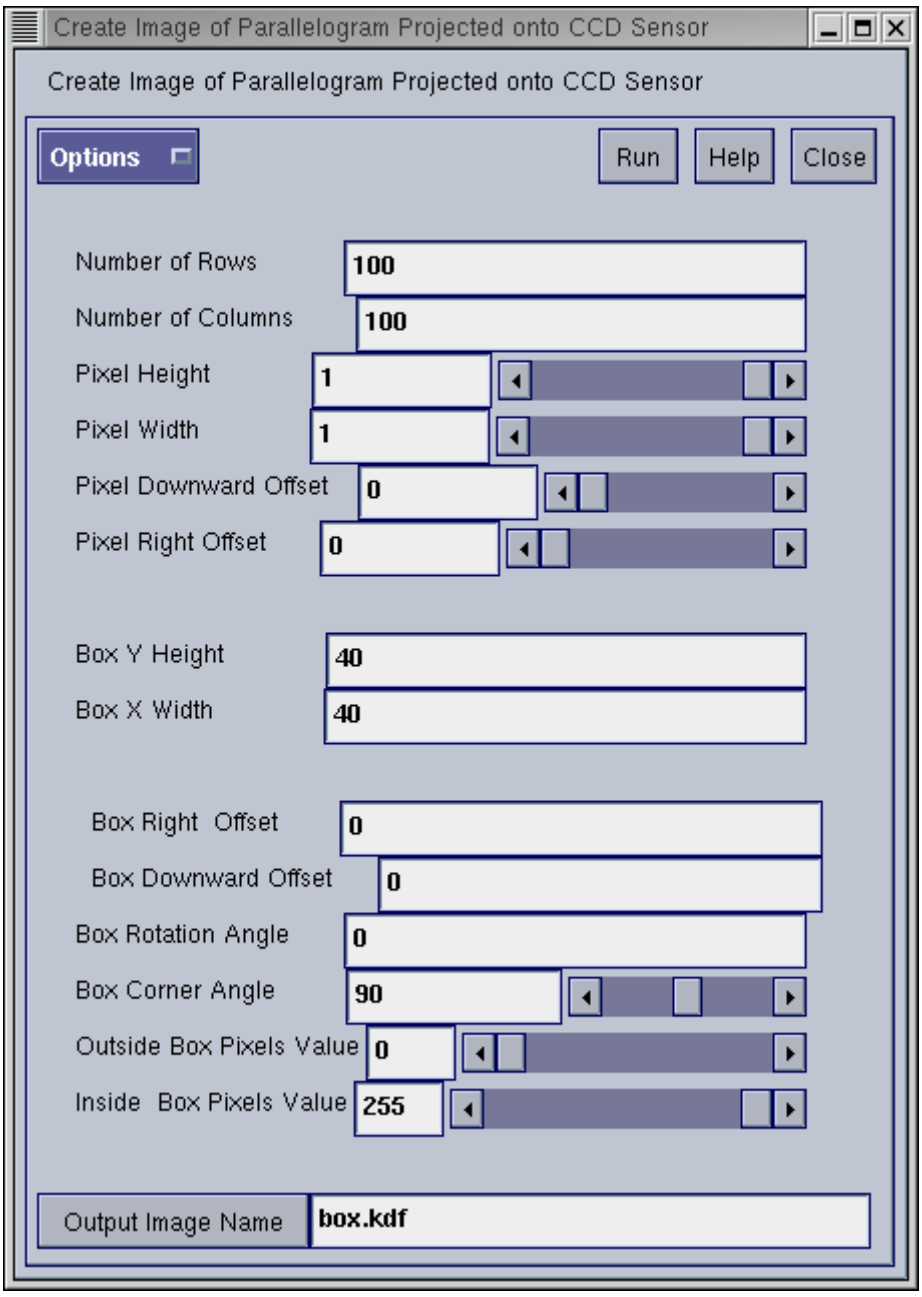
Prikazan je sadržaj dobivene datoteke "sin.ascii".

```
0
0.125333
0.24869
0.368125
0.481754
0.587785
0.684547
0.770513
0.844328
0.904827
0.951057
0.982287
...
```

2.4.2. Generiranje slike

U primjeru koji slijedi generirat će se siva slika pravokutnika pomoću igbox operatora.

```
%igbox -gui &
```



Slika 2.24. Prozor s parametrima operatora igbox.

Potrebno je postaviti parametare kao na slici 2.24, te upisati ime "box.kdf" za izlaznu datoteku. Klikom na [Run] generira se i izvršava u komandnoj liniji ekvivalentan igbox sa svim navedenim atributima.

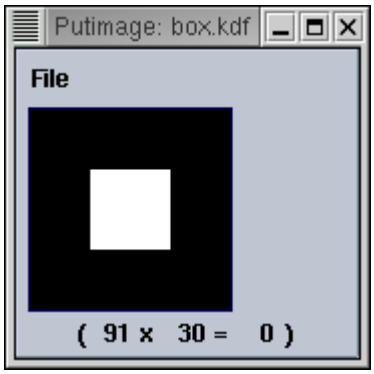
```
%igbox -r 100 -c 100 ... -h 40 -w 40 ... -o "box.kdf"
```

Opis zaglavlja dobivene datoteke se može vidjeti sa kfileinfo.

```
%kfileinfo –i box.kdf

# Name: box.kdf
# Comment:
# End Comment
# Machine Architecture: Little Endian IEEE
# Date Created: Wed Dec 11 13:10:50  2002
# Storage Format: kdf
# Format Description: Khoros 2.0 Data Format (kdf)
# Sub-Object Position:   0 , 0 , 0 , 0 , 0
# World Coordinate Point Size:   1 , 1 , 1 , 1 , 1
#
# Color Space Model:  0 (invalid)
# Has Alpha Channel:   0
#
# -- Value Data --
# Data Type: Unsigned Byte
# Size: Width = 100, Height = 100, Depth = 1, Time = 1, Elements = 1
```

Slika se može vidjeti pomoću putimage operatora.

```
%putimage –i box.kdf &
```



Slika 2.25. Slika prikazana operatorom putimage.

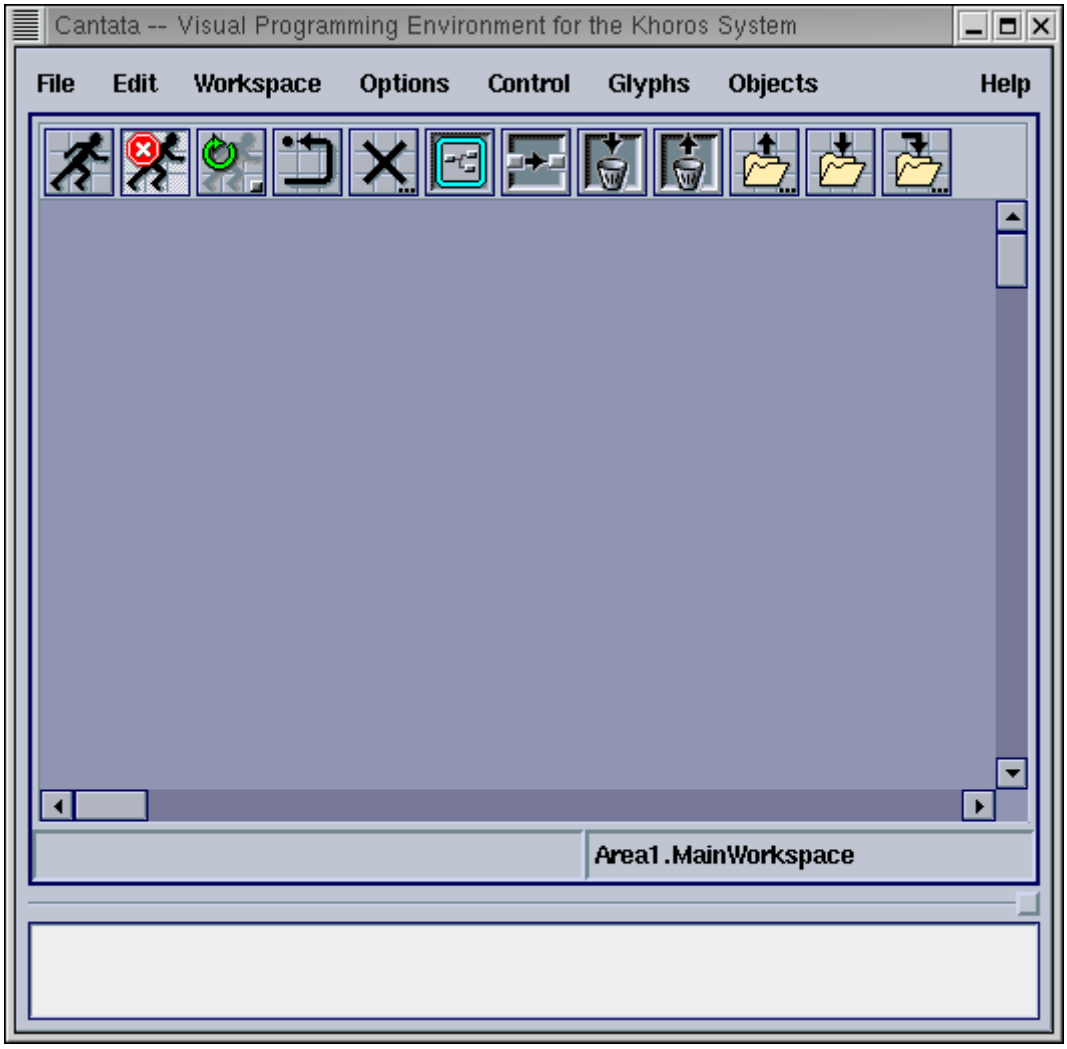
3. Cantata

Treće i najvažnije korisničko sučelje Khorosa je Cantata. To je okolina za vizualno programiranje, gdje su programi opisani grafovima u kojima operatori odgovaraju njihovim čvorovima. Cantata predstavlja savršen sustav za brzu izradu prototipova, testiranje i kreiranje novih operatora. Da bi se vidjela prednost Cantate u odnosu na CLUI/GUI sučelje, odnosno jednostavnost i brzina razvoja programa, ponovit će se primjeri pokazani ranije.

3.1. Opis Cantate i uvod u programiranje

Cantata se može pokrenuti na dva načina, prvo se pokrene Khoros, pa iz njegovog prozora odabere [Cantata], ili direktno iz komandne linije.

```
%cantata &
```



Slika 3.1. Grafička radna okolina.

Nakon pokretanja, prikaže se grafička radna okolina prikazana slikom 3.1.

Cantata sadrži nekoliko izbornika preko kojih korisnik može pristupiti različitim programima i alatima. Vrpca sa izbornicima nalazi se na krajnjem gornjem dijelu prozora.

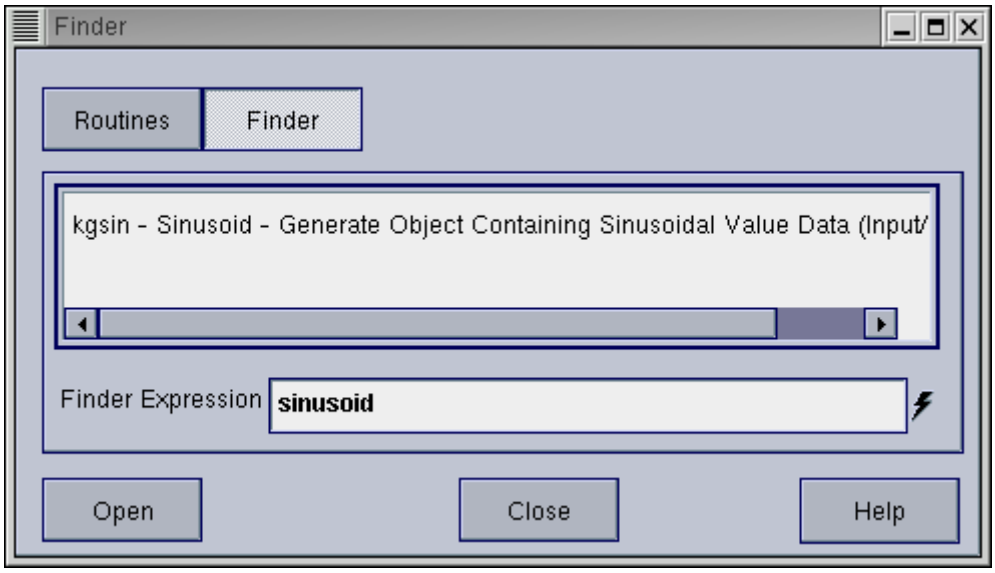
Ispod izbornika nalazi se vrpca sa ikonama koje predstavljaju najčešće korištene naredbe Cantate. Skup ikona je promjenjiv tako da korisnik može formirati vrpcu prema svojim potrebama.

Najveći središnji dio prozora je radni prostor na koji se postavlja mreža Glyphova.

Pri dnu prozora se nalazi tekstualna podloga na kojoj se za vrijeme izvođenja programa ispisuju ekvivalenti CLUI naredbi.

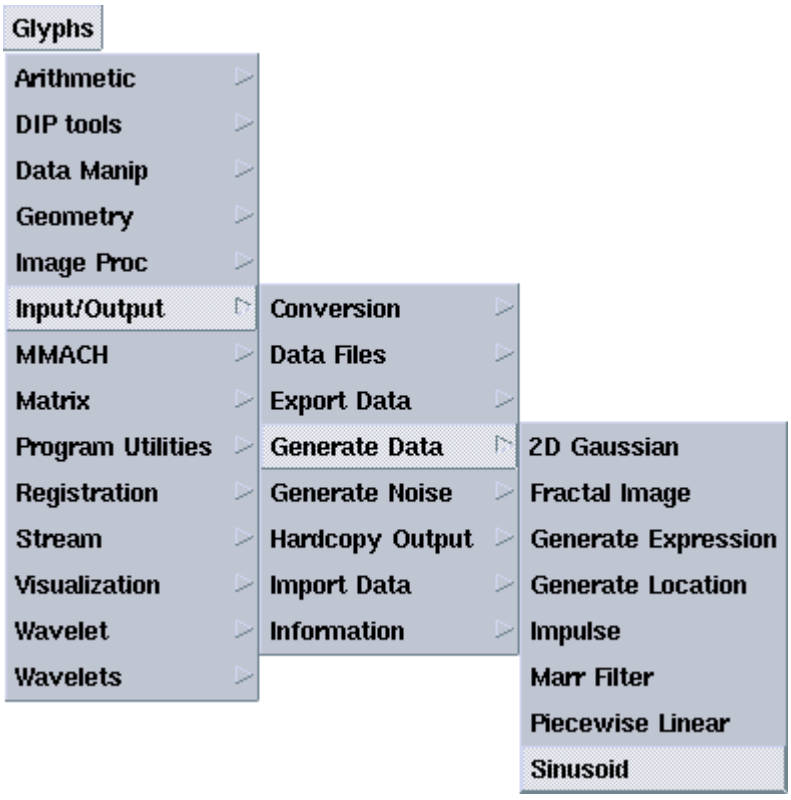
3.1.1. Generiranje i prikaz signala

Ponovit će se primjer generiranja sinusoide. Glyph koji se postavlja na radni prostor, može se pronaći pomoću alata za traženje [Edit] [Find...] [Finder]. Kao objekt traženja navede se "sinusoid". Rezultat traženja prikazan je slikom 3.2.



Slika 3.2. Pronađen je traženi Glyph.

Do njega se može doći i preko izbornika [Glyphs], kako je prikazano na slici 3.3.

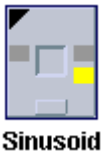


Slika 3.3. Odabir Glypha preko izbornika.

Nakon odabira Glypha, njegova kontura se pojavljuju na radnoj površini.

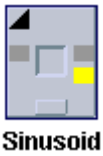
Slika 3.4. Kontura odabranog Glypha.

Kontura se može pomicati po radnoj površini, dok se na postavi na željeno mijesto. Nakon klika, kontura poprima stvarni izgled Glypha.

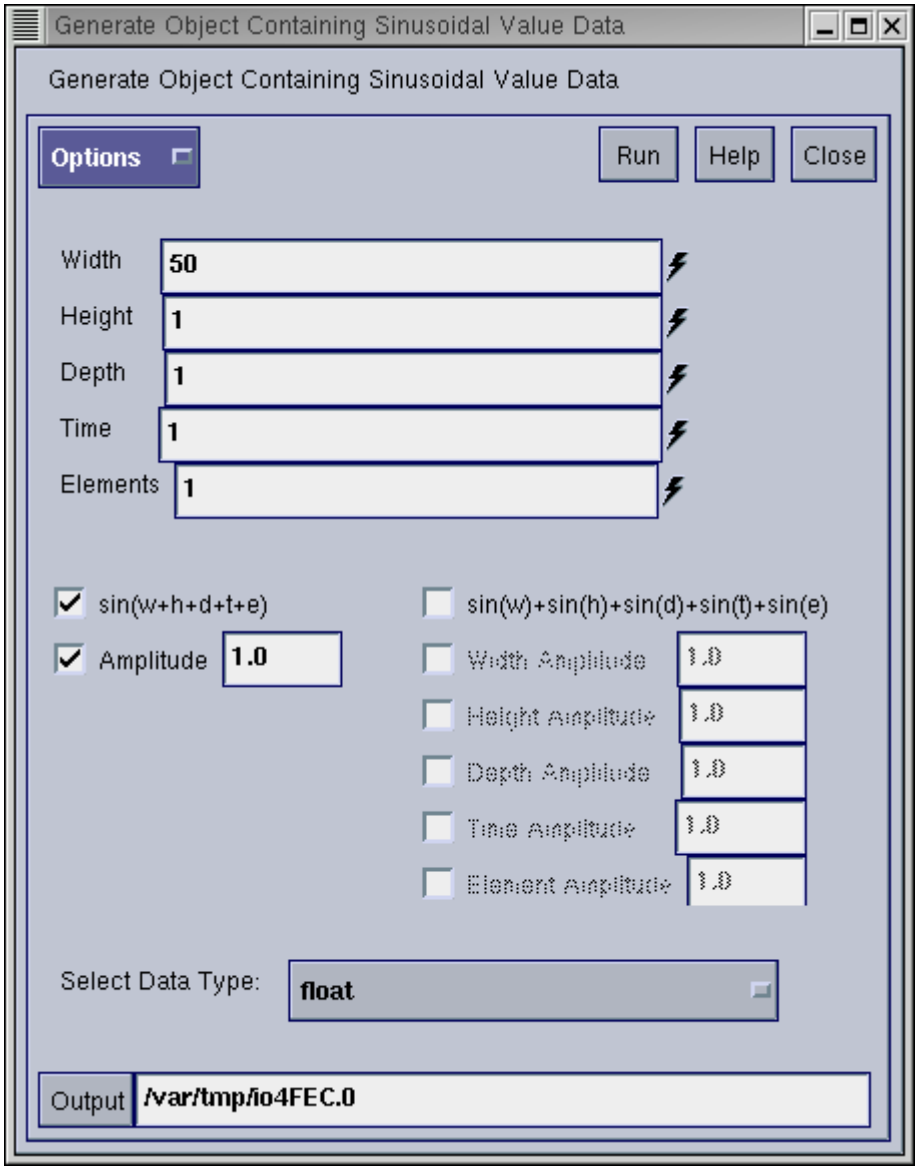


Slika 3.5. Glyph Sinusoid.

Pritiskom na trokut u gornjem lijevom uglu otvara se prozor sa parametrima koji je identičan onome pri pozivu kgsin operatora iz komandne linije sa argumentom "-gui". Izgled glypha za vrijeme otvorenog prozora prikazan je na slici 3.6.



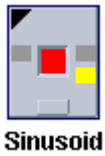
Slika 3.6. Glyph Sinusoid pri otvorenom parametarskom prozoru.



Slika 3.7. Parametarski prozor za Glyph Sinusoid.

Potrebno je postaviti parametare "Width" na 50, "Height" na 1, a ime ostaviti nepromjenjeno. Parametarski prozor se napušta odabirom [Close].

Operator se izvršava pritiskom na kvadratić u sredini Glypha. Za vrijeme izvođenja, taj kvadratić mijenjaja boju u crvenu, kao što je prikazano na slici 3.8.



Slika 3.8. Izgled Glypha za vrijeme izvođenja.

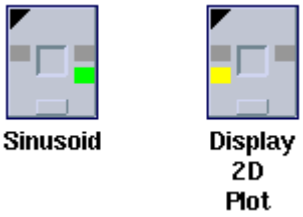
Pri dnu prozora Cantate, ispisuje se ekvivalentna naredba komandne linije.

```
# (new) DATAMANIP kgsin Sinusoid
%DATAMANIP/kgsin -wsize 50 -hsize 1 ... -o "/var/tmp/io4FEC.0" ...
```

Desni žuti kvadratić predstavlja izlaznu točku. Nakon završetka rada, Glyph ponovo poprima orginalni izgled, ali žuti kvadratić mijenja boju u zelenu, što znači da je operator napravio posao i da na izlazu postoji rezultat.

Slika 3.9. Izgled Glypha nakon obavljena posla.

Glyph za operator putplot2, može se naći pomoću alata za traženje. Do putplot2 se dolazi preko [Glyphs] [Visualization] [Plot Display] [Display 2D Plot]. Glyph se postavi na radnu površinu pokraj Glypha Sinusoid.



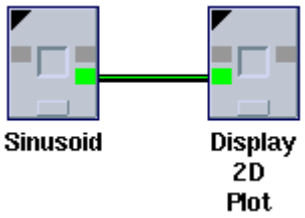
Slika 3.10. Dva Glypha prije povezivanja.

Žuti kvadratić Glypha operatora putplot2 predstavlja ulaznu točku. Žuta boja označava da je ulazna vrijednost obavezna, a kada je ulazna vrijednost opcionalna kvadratić je plave boje. Glyph Sinusoid nema ulaznu točku.

Na slici 3.11 je prikazan dio prozora sa parametrima Glypha Display 2D Plot, odnosno mjesto ulazne datoteke prije povezivanja sa glyphom Sinusoid.

Slika 3.11. Ulazna datoteka za operator putplot2.

Glyphovi se mogu povezati tako da se klikne na izlaznu točku jednog Glypha i ulaznu točku drugog Glypha, odnosno zeleni, pa zatim na žuti kvadratić. Povezivanje je moguće i obrnutim redom.



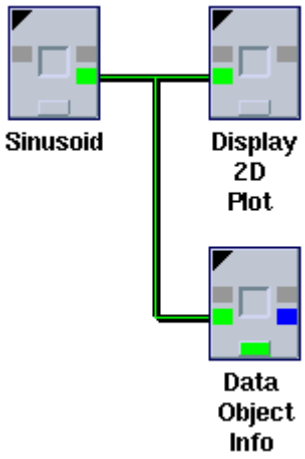
Slika 3.12. Povezani Glyphovi.

Kada se otvori prozor sa paremetrima za oba Glypha, može se vidjeti da je napravljeno povezivanje vrijednosti, odnosno da izlaz prvog Glypha predstavlja ulaz u drugi Glyph. Prikazan je dio prozora Glypha Display 2D Plot.

Slika 3.13. Ulazna datoteka za operator puplot2 nakon povezivanja.

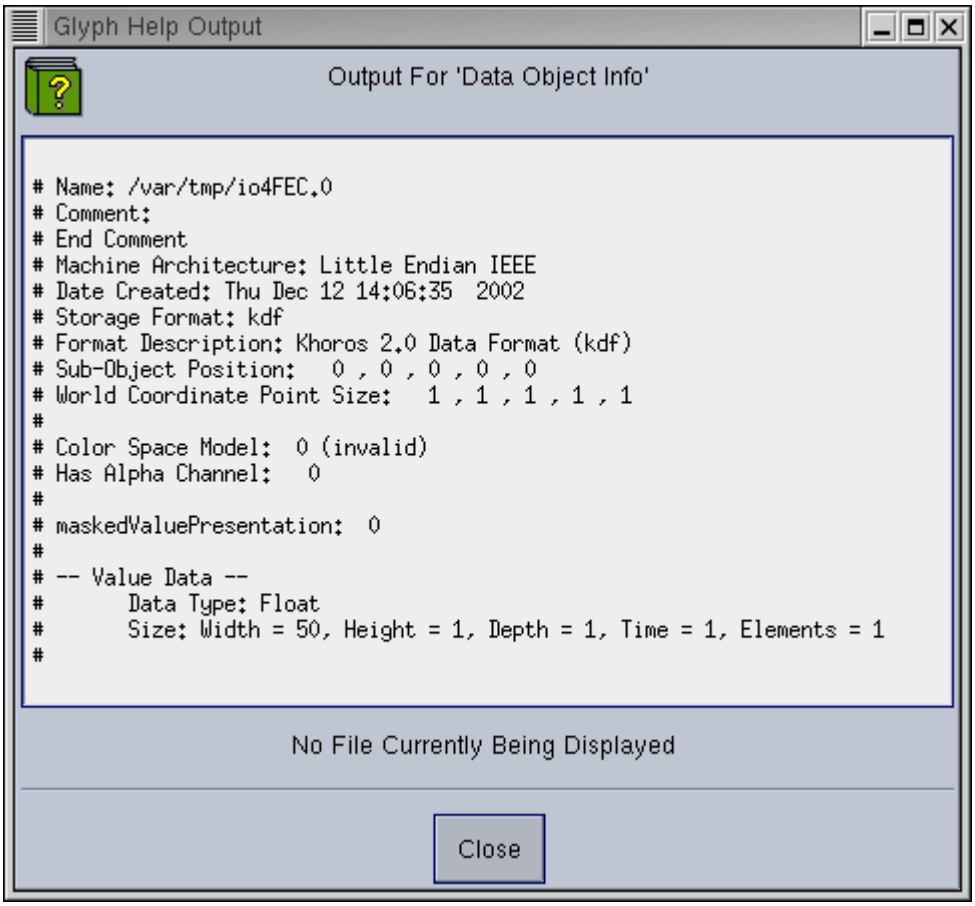
Pokretanjem Glypha Display 2D Plot dobije se slika sinusoide ista kao u primjeru iz komandne linije.

Glyph za kfileinfo, može se naći pomoću alata za traženje. Do njega se dolazi preko [Glyphs] [Input/Output] [Information] [Data Object Info]. Prikazano je kako bi sve trebalo izgledati nakon povezivanja trećeg Glypha.



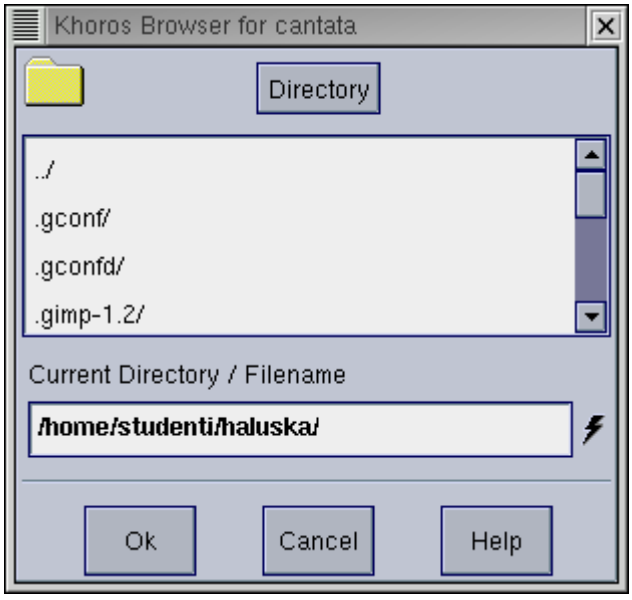
Slika 3.14. Povezana tri Glypha na radnoj površini.

Kvadratić pri dnu Glypha predstavlja standardni tekstualni izlaz, a kod trećeg Glypha je zelene boje jer je formirana poruka. Ako je crvene boje tada se radi o poruci greške. Poruka se može vidjeti klikom na zeleni kvadratić.



Slika 3.15. Izlazna poruka Glypha Data Object Info.

Formirana mreža Glyphova, odnosno program, može se pohraniti na disk odabirom [File] [Save File...].



Slika 3.16. Pohrana mreže Glyphova, odnosno programa na disk.

Uneseno ime datatoke dobiva ekstenziju ".wk" što je skraćenica za Workspace, ali korisnik može i sam koristiti svoju ekstenziju.

Iz Cantate se izlazi preko [File] [Exit].

3.1.2. Načini Rada

Izvođenje programa se može pokrenuti i bez klikanja na pojedine Glyphove, pomoću ikona za određivanje moda rada. Cantata može biti u dva osnovna stanja izvršavanja programa: Stop ili Run.

3.17. Načini rada, trenutno stanje je Stop.

Prva ikona označava Run, a druga Stop stanje. Slikom je prikazano Stop stanje. Pritiskom na drugu ikonu Cantata prelazi u Run stanje, odnosno izvršavaju se svi Glyphovi na radnoj površini.

3.18. Načini rada, trenutno stanje je Run.

Svaki Glyph se izvršava tek kad dobije ulazne vrijednosti. Nakon određenog vremena dođe se u stabilno stanje. Stabilno stanje se može poremetiti na više različitih načina, nakon čega dolazi do ponovnog izvršavanja programa. Razlog ponovnog izvršavanja može biti pokretanje pojedinog Glypha, mjenjanje njegovih parametara ili ulaznih vrijednosti. Tada se ponovno izvršavaju svi Glyphovi na koje promjena utječe, odnosno svaki Glyph povezan posredno ili neposredno sa izlaznom točkom Glypha u kojem se dogodila promjena.

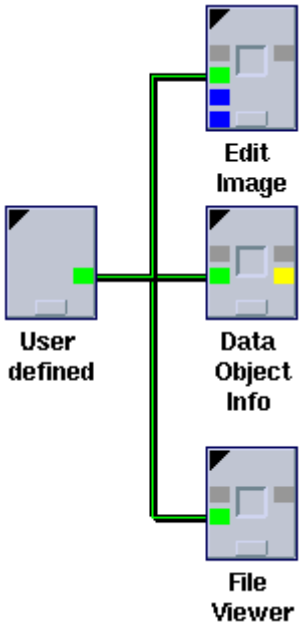
3.2. Rad sa različitim formatima podataka

Khoros podržava velik broj standardnih formata podataka. Operatori na ulazu razumiju podržane formate bez potrebe njihove pretvorbe. Nakon što pročita ulaznu datoteku Khoros prepozna format, pretvara datoteku u polimorfni model. Operatori obično na izlazu podatke pohranjuju u KDF formatu.

3.2.1. Pretvorba ASCII formata u KDF

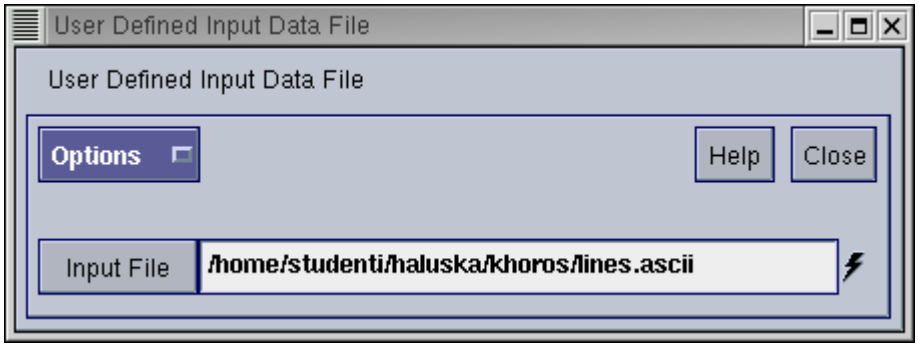
Pokazati će se pretvorba ASCII datoteke u KDF format, ali za razliku od primjera pod komandnom linijom, ovdje će se koristiti Cantata.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Data Files] [User Defined],
[Glyphs] [Visualization] [Interactive Image Display] [Edit Image],
[Glyphs] [Input/Output] [Information] [Data Object Info],
[Glyphs] [Input/Output] [Information] [File Viewer].



Slika 3.19. Program za pretvorbu ASCII datoteke.

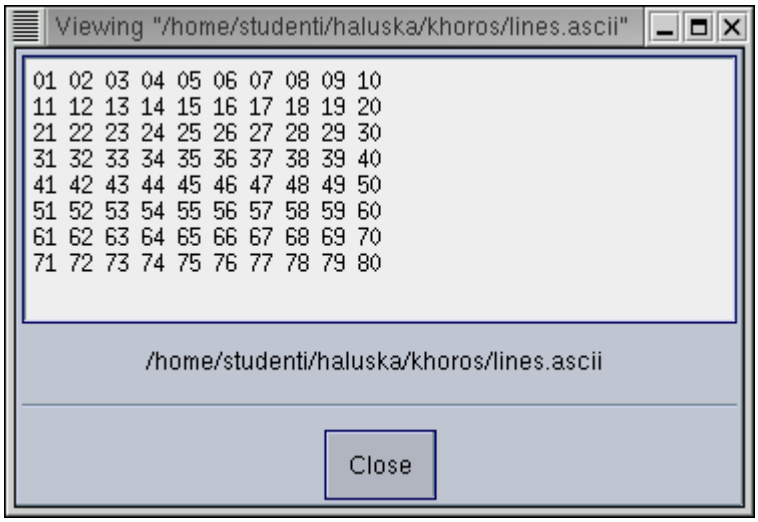
Glyph User defined predstavlja ulaznu točku u program, odnosno vezu sa ulaznom datotekom koja se nalazi u nekom formatu koji poznaje Khoros. Kod parametara za Glyph User defined potrebno je navesti ulaznu datoteku kao npr. "lines.ascii".



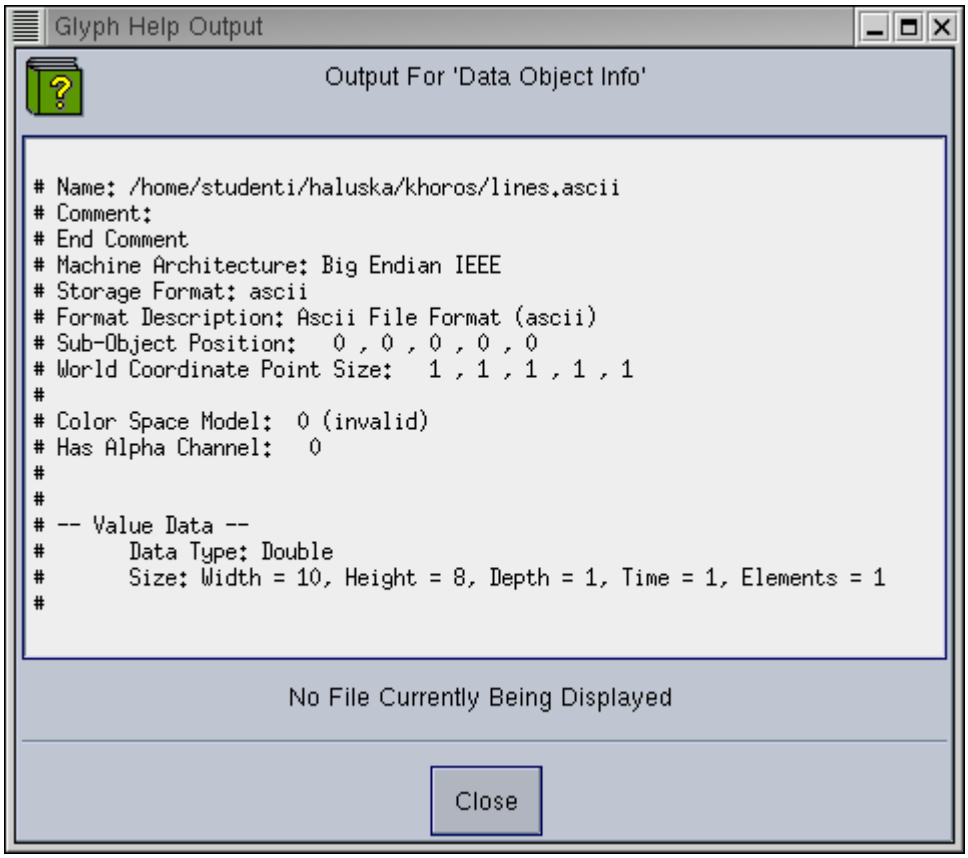
Slika 3.20. Navođenje imena ulazne ASCII datoteke.

Glyph Edit Image predstavlja operator editimage, a Glyph File Viewer služi za pregled tekstualne datoteke.

Nakon pokretanja programa dobivaju se rezultati isti kao kod pretvorbe iz komandne linije.



Slika 3.21. Sadržaj datoteke "lines.ascii".



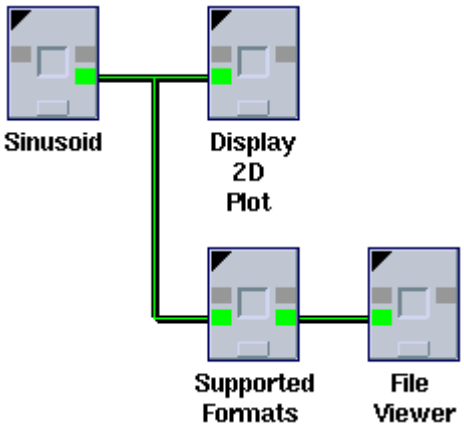
Slika 3.22. Izlazna poruka Glypha Data Object Info.

Osim navedenih prozora otvori se Editimage isti kao u primjeru iz komandne linije.

3.2.2. Pretvorba sinusoide u ASCII format

Pokazat će se pretvorba sinusoide u ASCII datoteku, ali za razliku od primjera pod komandnom linijom, ovdje će se koristiti Cantata.

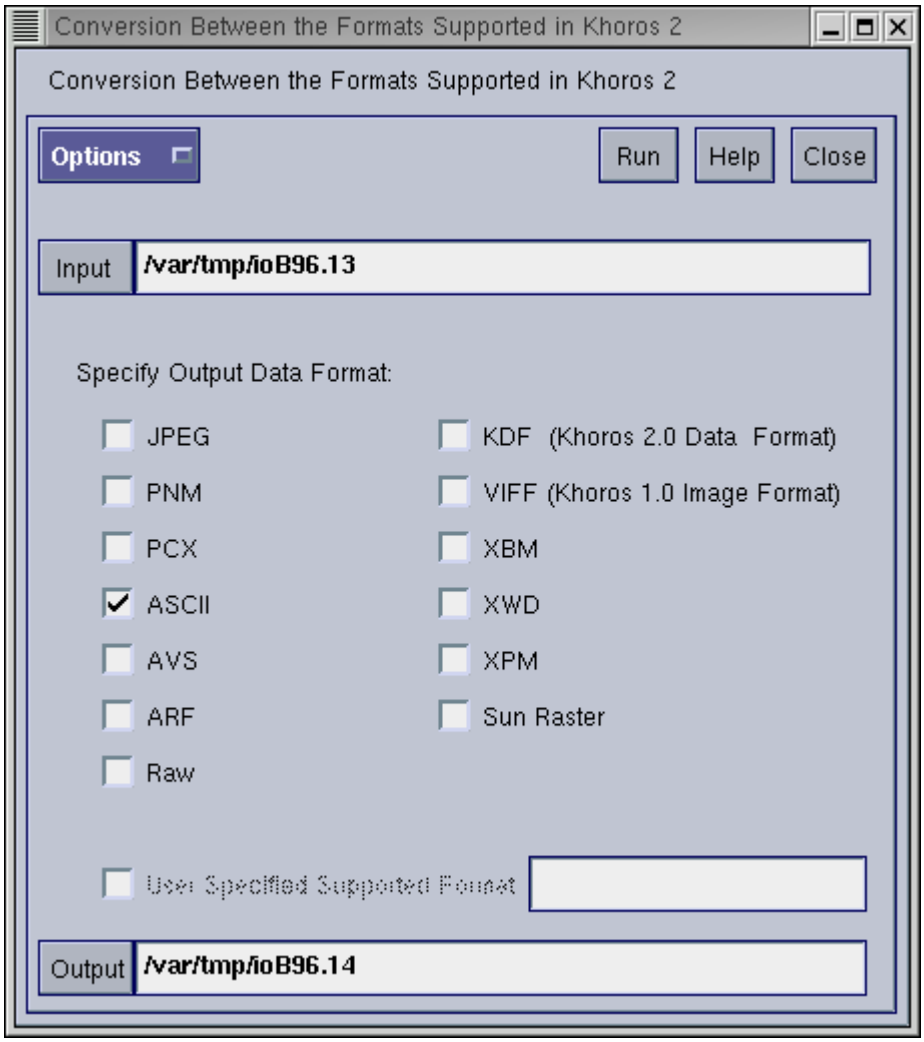
Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Generate Data] [Sinusoid],
[Glyphs] [Visualization] [Plot Display] [Display 2D Plot],
[Glyphs] [Input/Output] [Export Data] [Supported Formats],
[Glyphs] [Input/Output] [Information] [File Viewer].



Slika 3.23. Program za pretvorbu u ASCII format.

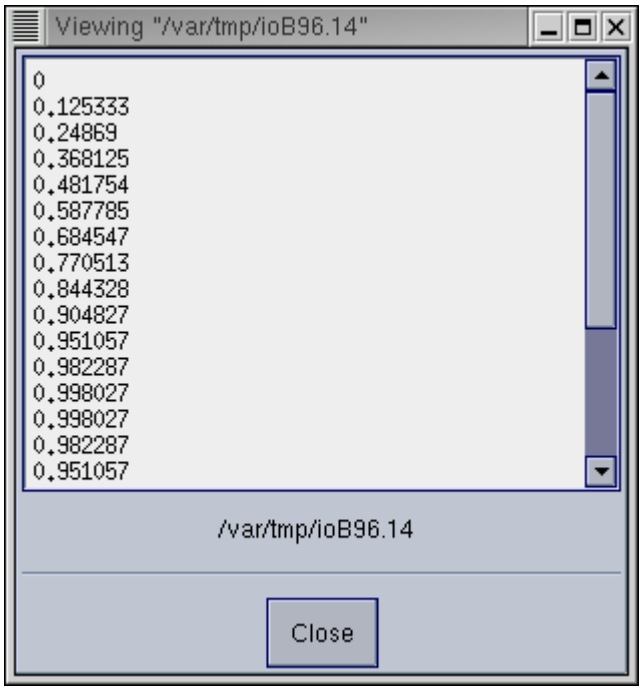
Kod Glypha Sinusoid potrebno je postaviti parametare "Width" na 50, "Height" na 1.

Glyph Supported Formats se upotrebljava za pretvorbu u neki od Khorosovih izlaznih formata. Potrebno je kod parametara kao izlazni format odrediti ASCII.



Slika 3.24. Odabir izlaznog formata.

Nakon pokretanja programa dobivaju se rezultati isti kao kod pretvorbe iz komandne linije.



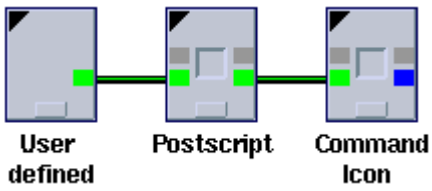
Slika 3.25. Dobivena ASCII datoteka.

Osim prozora sa sadržajem ASCII datoteke otvori se Editimage isti kao u primjeru iz komandne linije.

3.2.3. Pretvorba u PS format

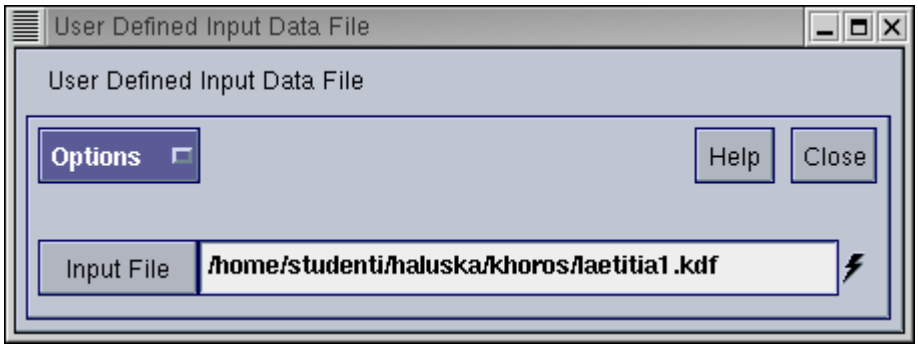
PS je format namjenjen ispisu, odnosno za prikaz rezultata na pisaču, pa ga nema smisla koristiti za razmjenu podataka. Khoros sadrži operator za pretvorbu slika u PS format.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Data Files] [User Defined],
[Glyphs] [Input/Output] [Hardcopy Output] [Postscript],
[Glyphs] [Program Utilities] [General] [Command Icon].



Slika 3.26. Program za pretvorbu u PS format.

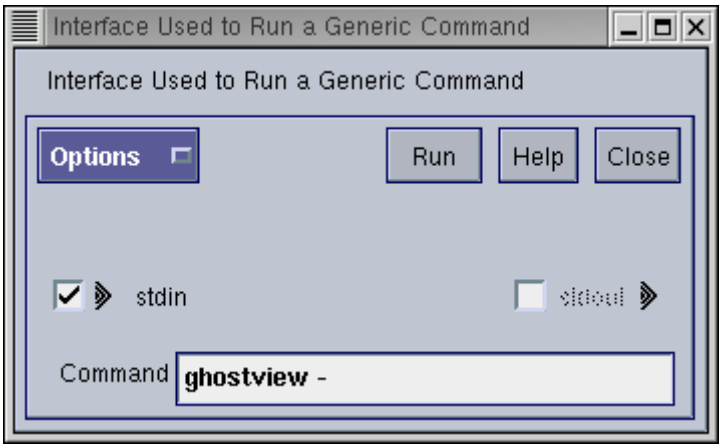
Za ulaznu datoteku pod parametrima Glypha User defined odabere se "laetitia1.kdf".



Slika 3.27. Određivanje ulazne datoteke.

Glyph Postscript se koristi za pretvorbu u PS format.

Command Icon predstavlja sučelje za poziv vanjskih programa koji koriste standardne stdin i stdout veze. U ovom primjeru Glyph predaje rezultat u PS formatu programu ghostview. Pa se kod parametara Glypha navede se ghostview kao ime programa koji će otvoriti izlaznu PS datoteku.



Slika 3.28. Određivanje parametara za Glyph Command Icon.

Kao rezultat programa otvara se prozor programa ghostview sa PS slikom.

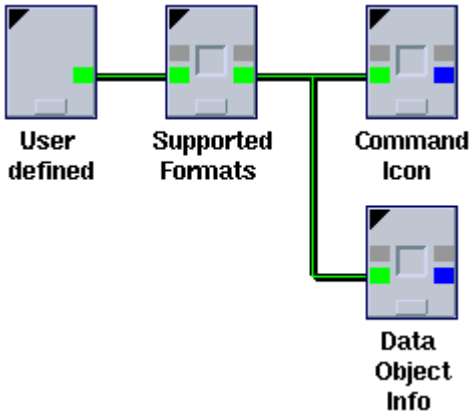
Slika 3.29. Dobivena slika prikazana sa programom ghostview.

3.2.4. Pretvorba u formate koje Khoros ne podržava

Formati koje Khoros podržava su navedeni ranije. Khoros ne podržava sve formate, ali postoji način pretvorbe preko sučelja Cantate, koristeći vanjske programe za prikaz slike koji podržavaju tražene formate.

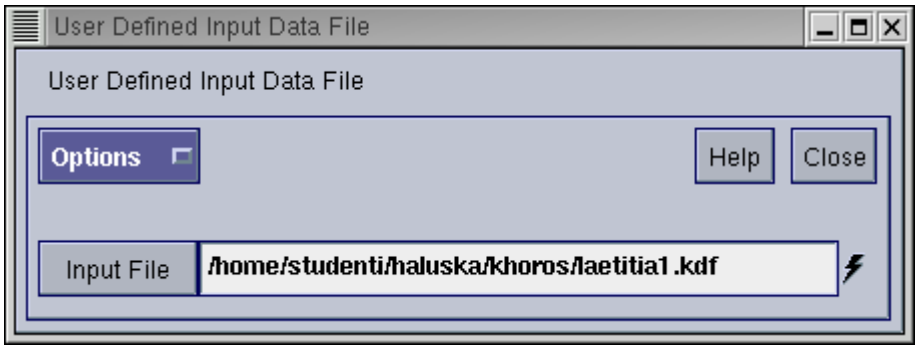
Slika "laetitia1.kdf" će se pretvoriti u PNM format, te predati xv, poznatom programu za prikaz slike koji podržava gotovo sve slikovne formate.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Data Files] [User Defined],
[Glyphs] [Input/Output] [Export Data] [Supported Formats],
[Glyphs] [Program Utilities] [General] [Command Icon],
[Glyphs] [Input/Output] [Information] [Data Object Info].



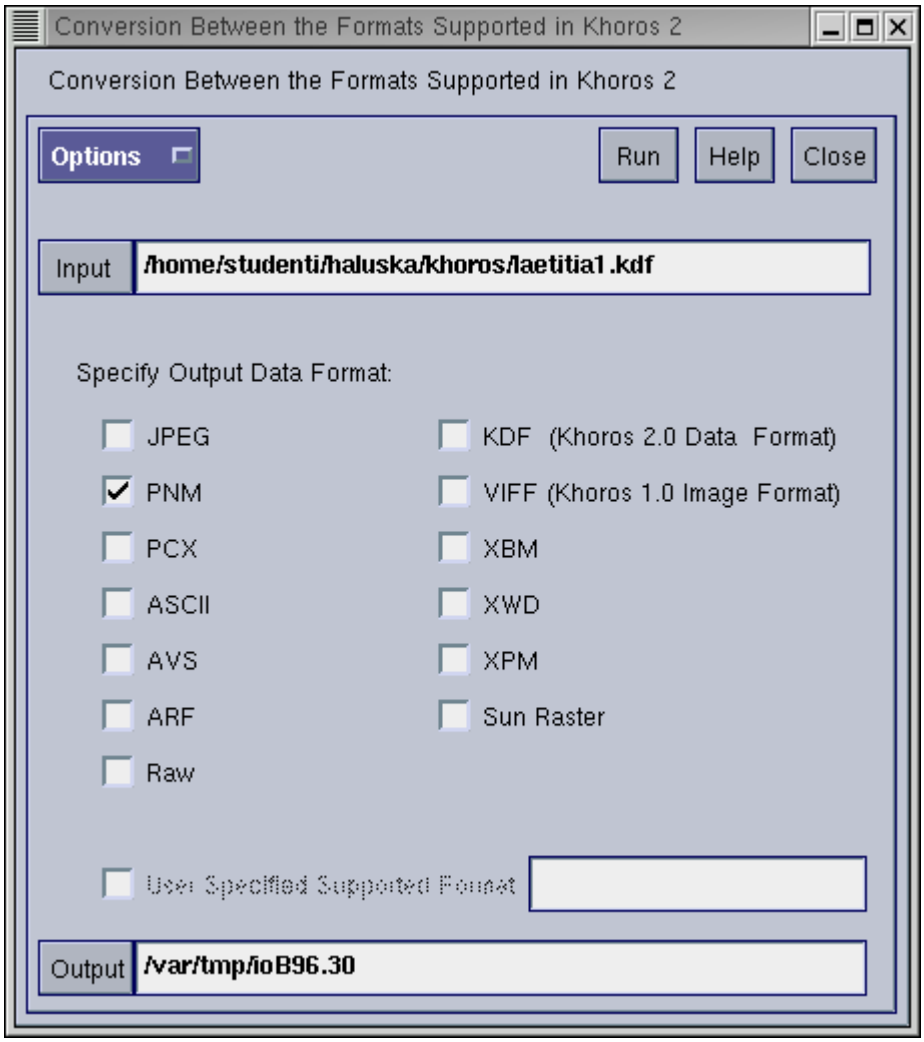
Slika 3.30. Program za pretvorbu u PNM format.

Kod parametara Glypha User defined pootrebno je navesti ulaznu datoteku "laetitia1.kdf".



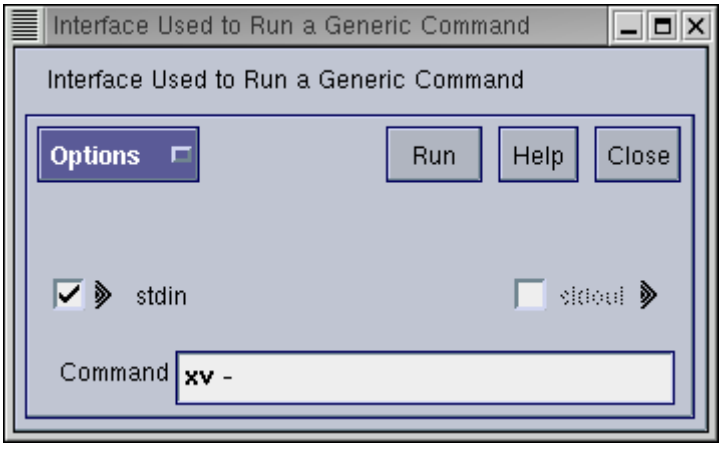
Slika 3.31. Određivanje ulazne datoteke.

Glyph Supported Formats pretvara ulaznu datoteku u jedan od formata koji podržava Khoros. U ovom primjeru datoteka će biti pretvorena u PNM format, što je potrebno navesti kod parametara Glypha.



Slika 3.32. Određivanje izlaznog formata datoteke.

Kod parametara Glypha Command Icon navede se xv kao ime programa koji će otvoriti izlaznu PNM datoteku.



Slika 3.33. Određivanje parametara za Glyph Command Icon.

Kao rezultat otvara se prozor programa xv sa PNM slikom.

Slika 3.34. PNM slika prikazana sa programom xv.

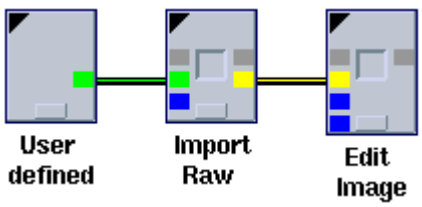
Pomoću programa xv slika se može pohraniti u bilo kojem formatu koji podržava program xv.

Drugi programi poput xv, mogu de koristiti i za ulaznu pretvorbu formata. Tada se slika pohranjena u formatu koji Khoros ne razumije, otvori pomoću xv, ili nekog drugog programa i pohrani u jedan od formata koje Khoros poznaje.

3.2.5. Rad sa slikama nepoznatog formata

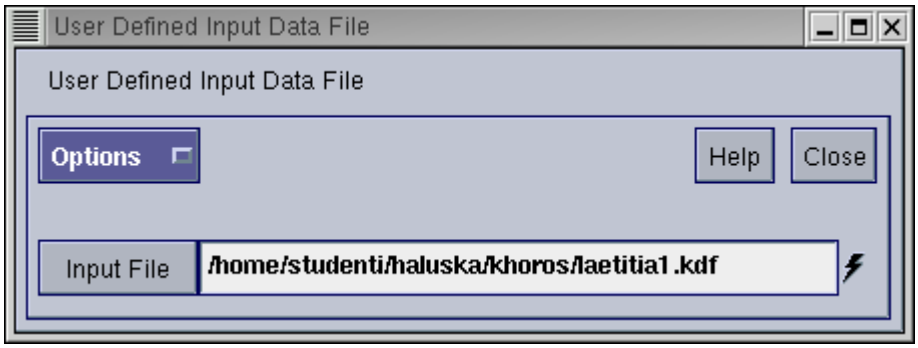
Kada je potrebno kao ulaznu datoteku koristiti sliku nepoznatog formata, Cantata pomoću Glypha Import Raw omogućuje da se putem pokušaja/pogrešaka jednostavnija slika ipak prepozna.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Data Files] [User Defined],
[Glyphs] [Input/Output] [Import Data] [Import Raw],
[Glyphs] [Visualization] [Interactive Image Display] [Edit Image].



Slika 3.35. Program za rad sa nepoznatim ulaznim formatom.

Kao slika nepoznatog formata, pokušat će se prepoznati "laetitia1.kdf", pa je pod parametrima Glypha User defined treba navesti kao ulaznu datoteku.

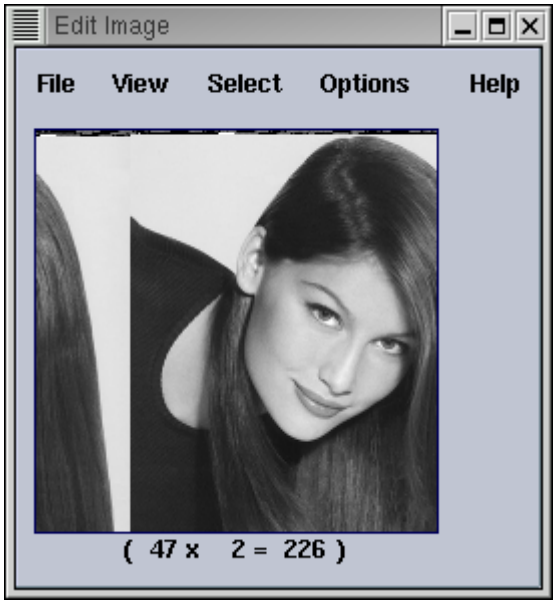


Slika 3.36. Određivanje ulazne datoteke.

Neka se pretpostavi da su dimenzije ulazne slike 200 x 200 točaka, pa se to i navede kod parametara Glypha Import Raw.

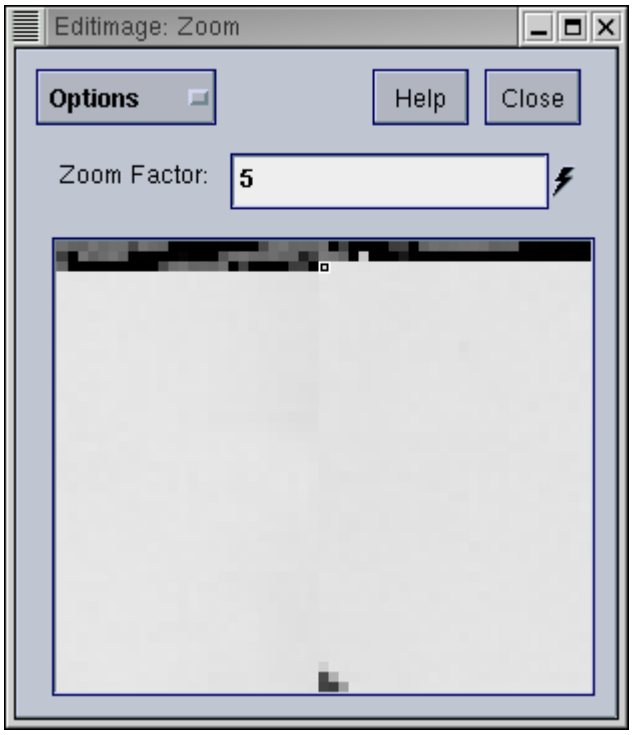
Slika 3.37. Određivanje dimenzija slike.

Kao rezultat dobije se slika sa pomakom.

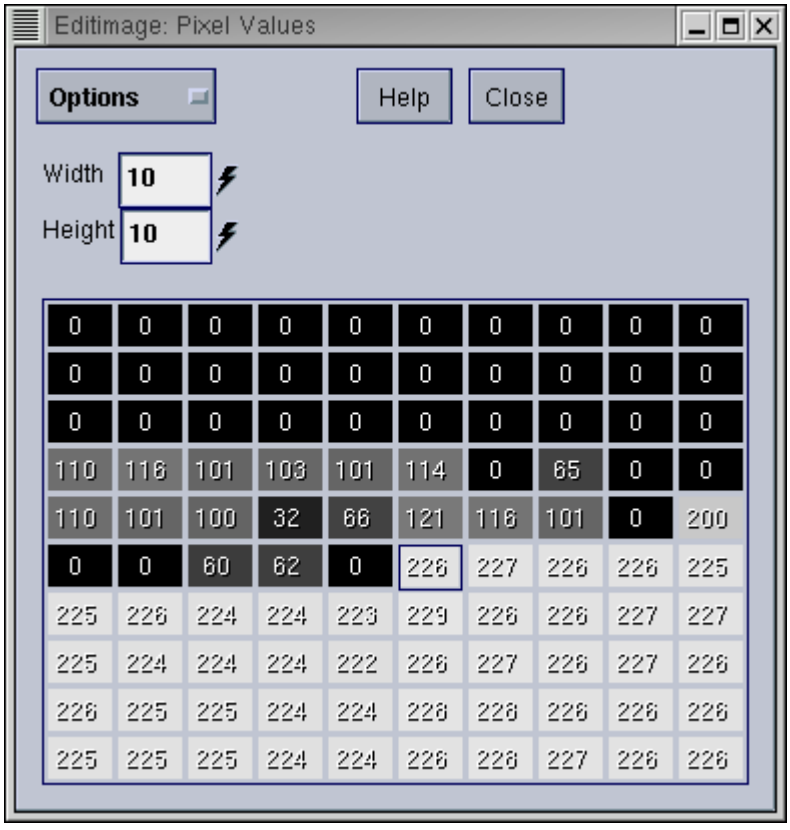


Slika 3.38. Početna slika sa pomakom.

Da bi se vidjelo gdje završava zaglavlje, a počinje slika otvore se pomoćni prozori Editimage operatora.



Slika 3.39. Zoom prozor za određivanje početka slike.

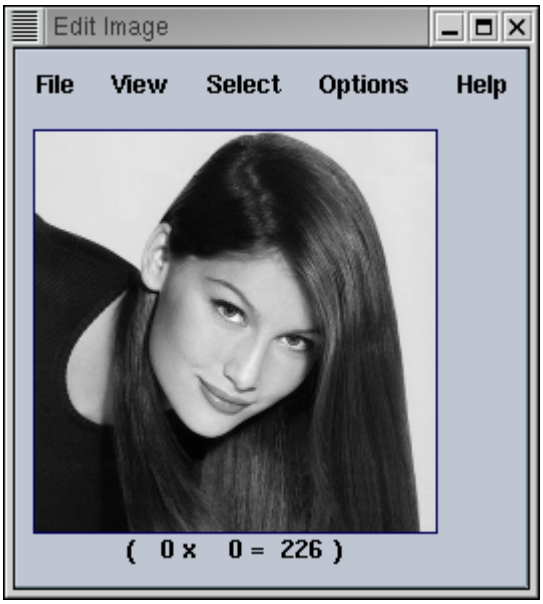


Slika 3.40. Pixel Values prozor za određivanje početka slike.

Iz pomoćnih prozora se odredi točan početak slike, na 2*200+47=247, pa se vrijednost 247 unese kod parametara Glypha Import Raw.

Slika 3.41. Unos parametra Offset.

Nakon ponovnog pokretanja programa, slika je u redu, bez pomaka i niza tamnih točaka pri vrhu.



Slika 3.42. Slika polako poprima orginalni izgled.

Vrijednost slikovnog elementa (0, 0) je 226, točno kao što treba biti, što se vidi iz prijašnjeg Pixel Values prozora. Da bi se odredile konačne dimenzije slike pokušava se povećavati visina na 300.

Slika 3.43. Prevelika visina slike.

Slično kao za početnu sliku, pomoću Zoom i Pixel Values prozora za sliku sa visinom 300, izračuna se točna visina, 300-20=280. Konačno, nakon unosa sa visinu vrijednosti 280, slika poprima orginalne dimenzije.

3.3. Polimorfni model podataka

U ranije pokazanim primjerima jedino je korišten vrijednosni segment polimorfnog modela podataka. Ovdje će se pokazati primjeri vezani za mapu.

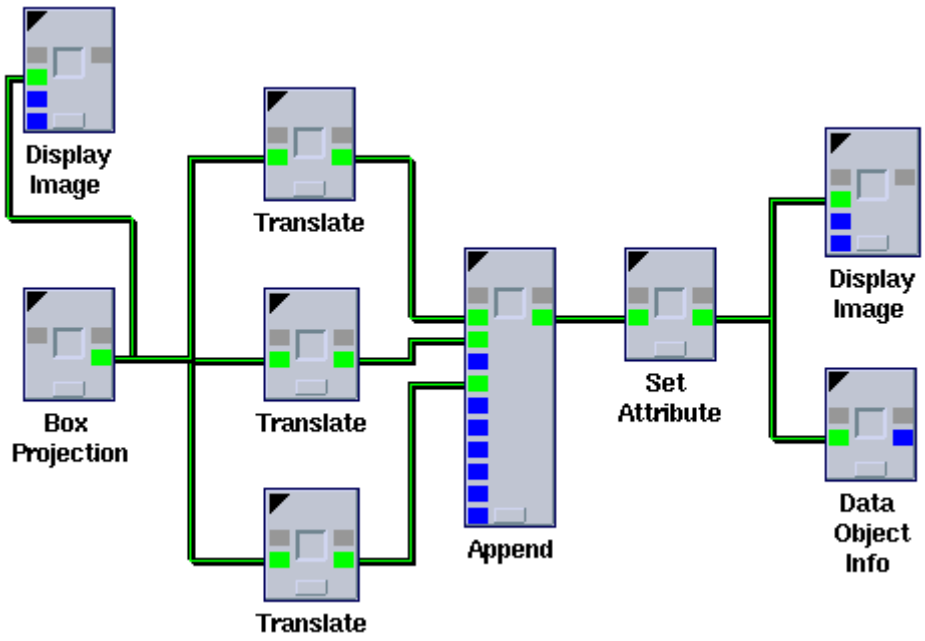
3.3.1. RGB model

Slike u boji se mogu prikazati pomoću više modela. Jedan od tih modela je RGB, gdje je svaki slikovni element sastavljen od tri komponente: crvene, zelene i plave. Komponente međusobnom kombinacijom generiraju ostale boje.

U Khorosovom polimorfnom modelu podataka komponente boje se pohranjuju u dimenziji elemenata. Element 0 odgovara crvenoj, element 1 odgovara zelenoj i element 2 plavoj.

Primjerom će se pokazati formiranje RGB slike.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Conversion] [Box Projection],
[Glyphs] [Visualization] [Non-Interactive Image Display] [Display Image],
[Glyphs] [Data Manip] [Reorganize Data] [Translate],
[Glyphs] [Data Manip] [Size & Region Operators] [Append],
[Glyphs] [Data Manip] [Object Attributes] [Set Attribute],
[Glyphs] [Input/Output] [Information] [Data Object Info].



Slika 3.44. Program za formiranje RGB slike.

Glyph Box Projection generira sliku pravokutnika. Dimenzije slike i dimenzije pravokutnika se navedu u parametarskom prozoru.

Number of Rows	100
Number of Columns	100
Box Y Height	40
Box X Width	40

Slika 3.45. Definiranje parametara za sliku pravokutnika.

Glyph Translate translatira sliku, odnosno u ovom primjeru pravokutnike pomiče za navedene vrijednosti. Potrebno je u parametarske prozore unijeti vrijednosti prikazane slikom.

☑ Width Offset	10
☑ Height Offset	10

Slika 3.46. Parametri za prvi Translate Glyph.

Slika 3.47. Parametri za drugi Translate Glyph.

Slika 3.48. Parametri za treći Translate Glyph.

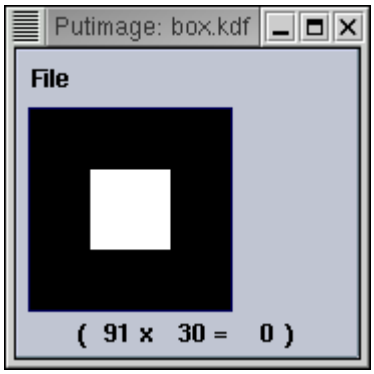
Glyph Append spaja translatirane slike, tako da svaku stavlja u drugu dimenziju elemenata. Potrebno je kod parametara navesti da se objekti slažu duž dimenzije elemenata.

Slika 3.49. Parametri za Glyph Append.

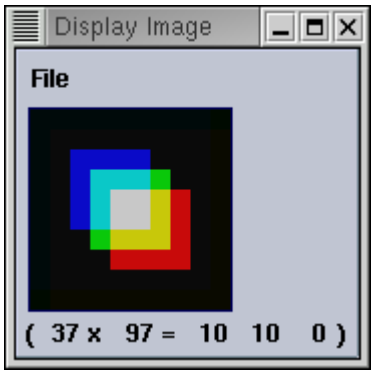
Da bi se podaci koji se nalaze kroz dimenziju elemenata interpretirali kao RGB slika, koristi se Glyph Set Attribute kod kojeg se preko parametara određuje RGB model.

Slika 3.50. Parametri za Glyph Set Attribute.

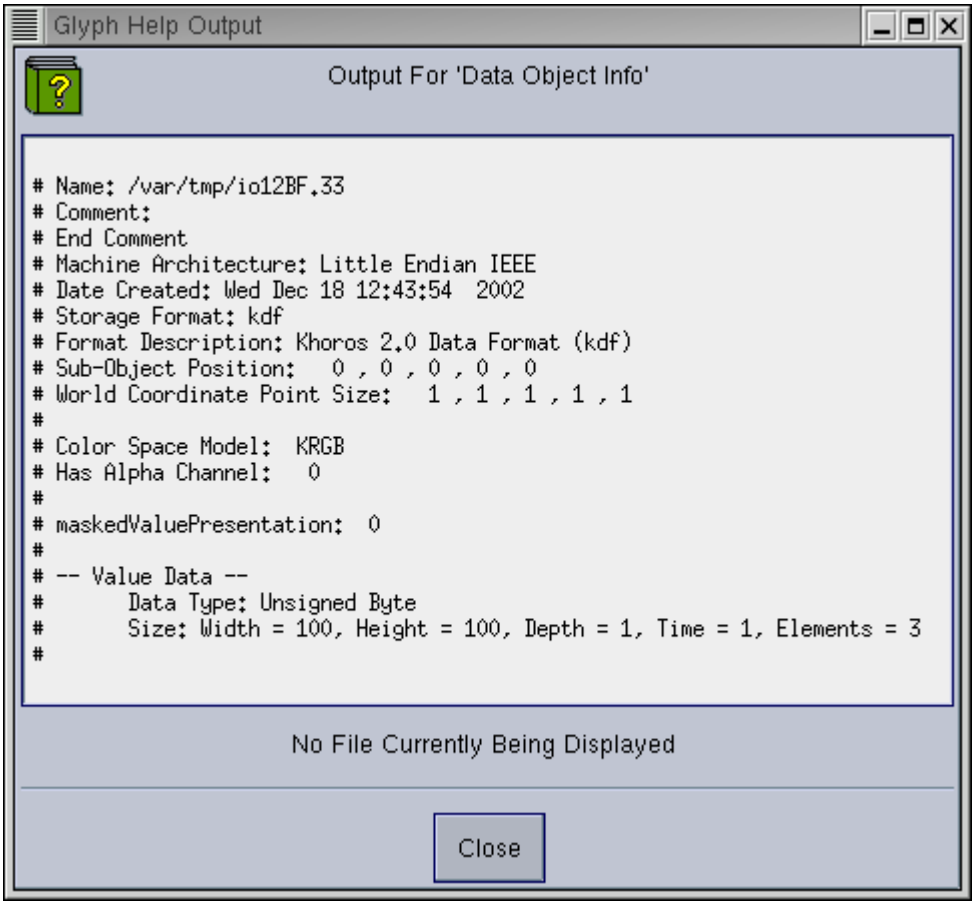
Rezultat izvođenja programa prikazan je slikama.



Slika 3.51. Početna slika.



Slika 3.52. Slika dobivena kao rezultat programa.

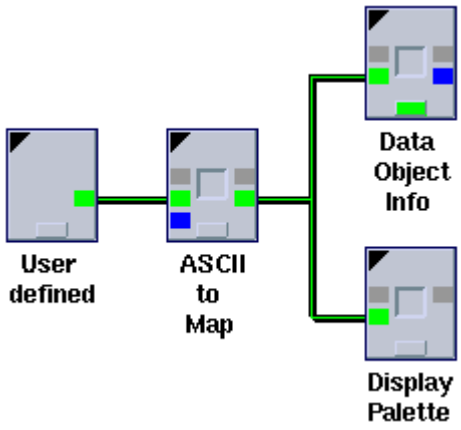


Slika 3.53. Data Object Info dobivene slike.

3.3.2. Formiranje mape i unos u sliku

Ako se želi formirati RGB slika sa indeksom na mapu, potrebno je formirati ASCII datoteku boja koja će se koristiti kao mapa slike, te ju na kraju dodati slici sa sivim razinama.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Data Files] [User Defined],
[Glyphs] [Input/Output] [Import Data] [ASCII to Map],
[Glyphs] [Input/Output] [Information] [Data Object Info],
[Glyphs] [Visualization] [Color Map Display & Manipulation] [Display Palette].

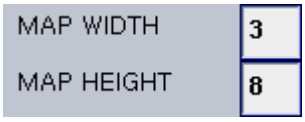


Slika 3.54. Program za formiranje mape.

Kao ulazna datoteka kod parametara Glypha User Defined navede se "map.ascii" koja sadrži RGB komponente za 8 boja.

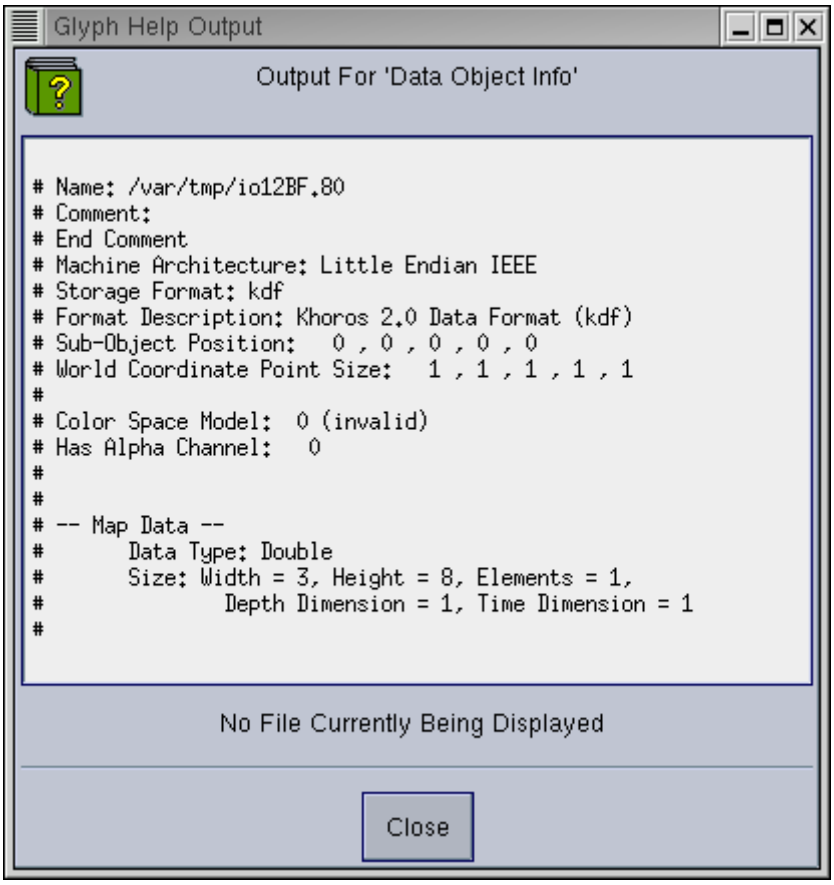
```
  0   0   0
255   0   0
  0 255   0
255 255   0
  0   0 255
255   0 255
  0 255 255
255 255 255
```

Glyph ASCII to Map pretvara ulaznu datoteku u mapu. Kod parametara Glypha potrebno je navesti dimenzije mape.

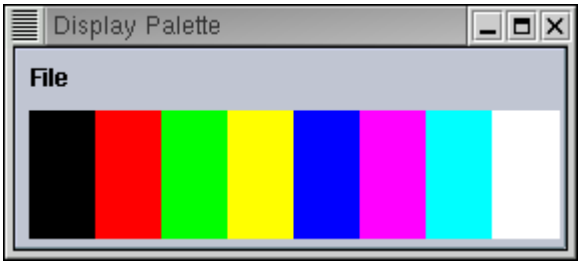


Slika 3.55. Parametri Glypha ASCII to Map.

Kao rezultat dobiva se mapa u Polimorfnom modelu podataka. Rezultati izvršavanja programa su prikazani slikama.

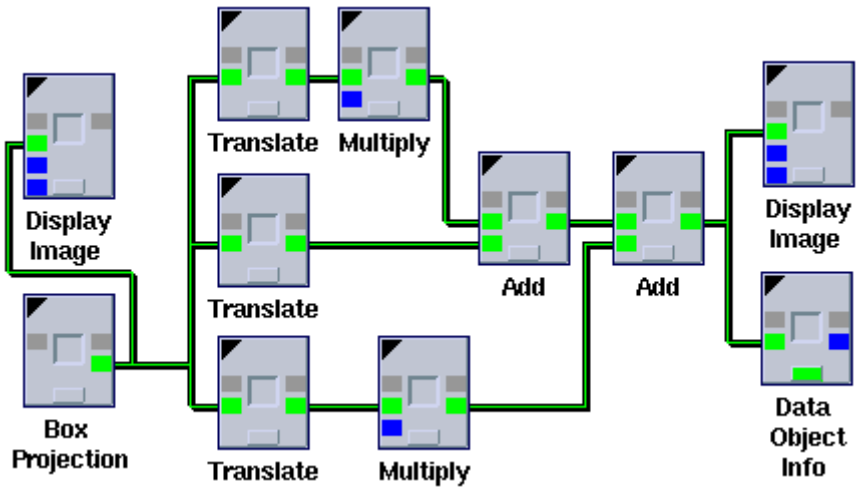


Slika 3.56. Poruka Glypha Data Object Info.



Slika 3.57. Izgled formirane RGB mape.

Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Conversion] [Box Projection],
[Glyphs] [Visualization] [Non-Interactive Image Display] [Display Image],
[Glyphs] [Data Manip] [Reorganize Data] [Translate],
[Glyphs] [Arithmetic] [Two Operand Arithmetic] [Multiply],
[Glyphs] [Arithmetic] [Two Operand Arithmetic] [Add],
[Glyphs] [Visualization] [Non-Interactive Image Display] [Display Image],
[Glyphs] [Input/Output] [Information] [Data Object Info].



Slika 3.58. Program za formiranje sive slike.

Parametri Glypha Translate se podese kao u prethodnom primjeru.

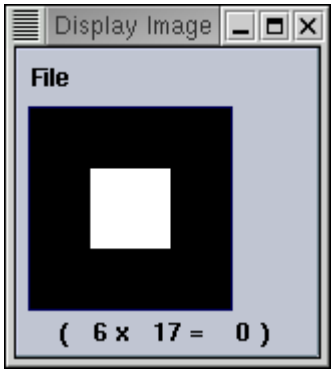
Glyph Multiply množi vrijednost svakog slikovnog elementa sa ulaznom vrijednošću ili konstantom. Kod parametara treba odabrati konstante 2 i 4, pa će se na izlazu dobiti sive slike sa vrijednostima sivih razina 1, 2 i 4.

Slika 3.59. Parametri za prvi Multiply Glyph.

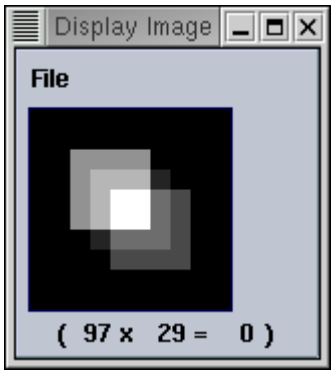
Slika 3.60. Parametri za drugi Multiply Glyph.

Glyph Add zbraja dva ulaza i na izlazu daje rezultat.

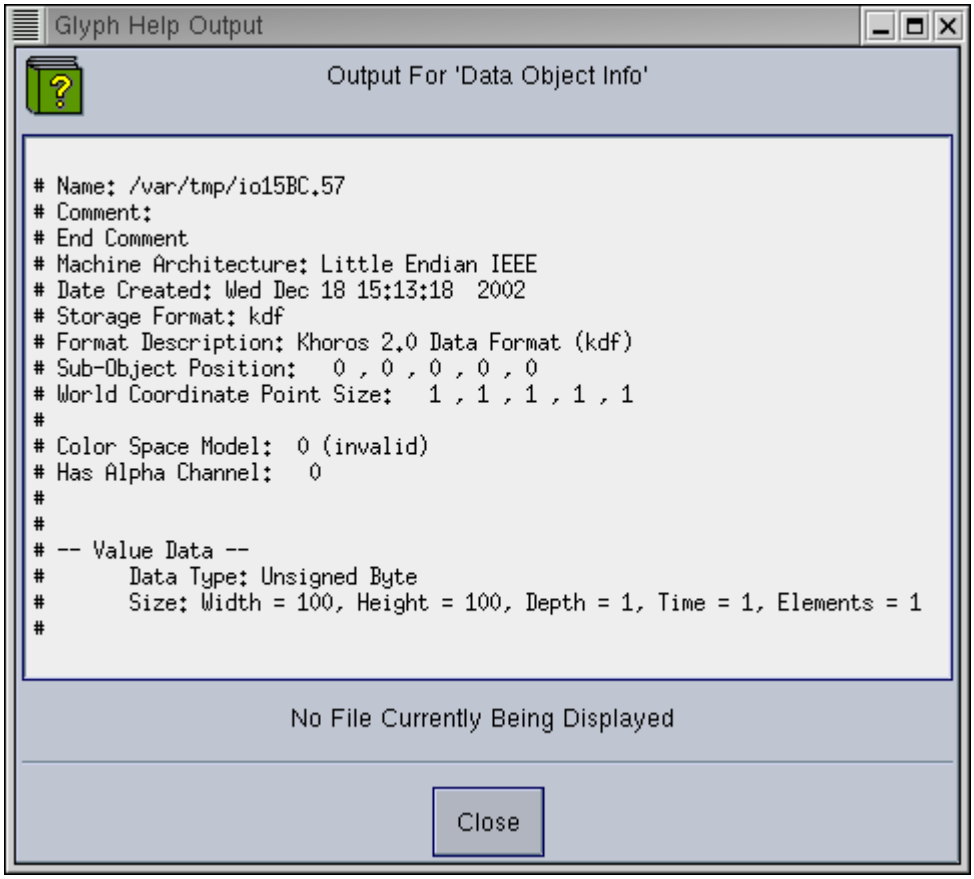
Rezultati izvršavanja programa su prikazani slikama.



Slika 3.61. Početna slika.

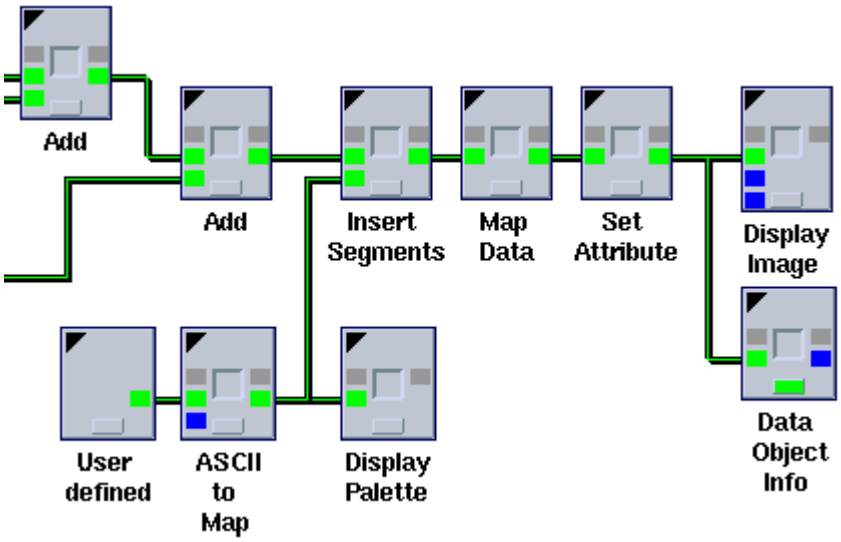


Slika 3.62. Slika dobivena kao rezultat programa.



Slika 3.63. Poruka Glypha Data Object Info.

Potrebno je povezati program za formiranje mape sa programom za formiranje sive slike. Način povezivanja i ostatak dodanih Glyphova prikazan je slikom. Dodani su Glyphovi:
[Glyphs] [Data Manip] [Segment Operators] [Insert Segments],
[Glyphs] [Data Manip] [Map Operators] [Map Data],
[Glyphs] [Data Manip] [Object Attributes] [Set Attribute].



Slika 3.64. Konačni izgled programa.

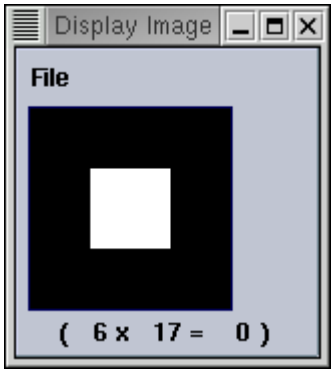
Glyph Insert Segments dvije ulazne slike, sivu sliku i mapu, pretvara u jednu, koja sadrži vrijednosni segment i mapu, odnosno indeksnu sliku u boji.

Glyph Map Data pretvara ulaznu indeksnu sliku u pravu sliku u boji, odnosno nestaje segment mape, a dimenzija elemenata se povećava sa 1 na 3.

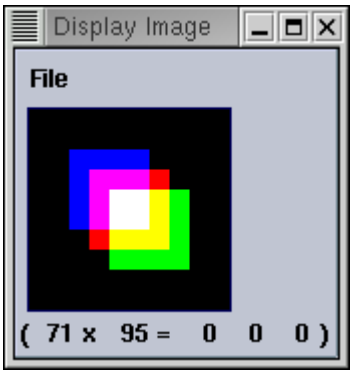
Da bi se podaci koji se nalaze kroz dimenziju elemenata interpretirali kao RGB slika, koristi se Glyph Set Attribute kod kojeg se preko parametara određuje RGB model.

Slika 3.65 Parametri za Glyph Set Attribute.

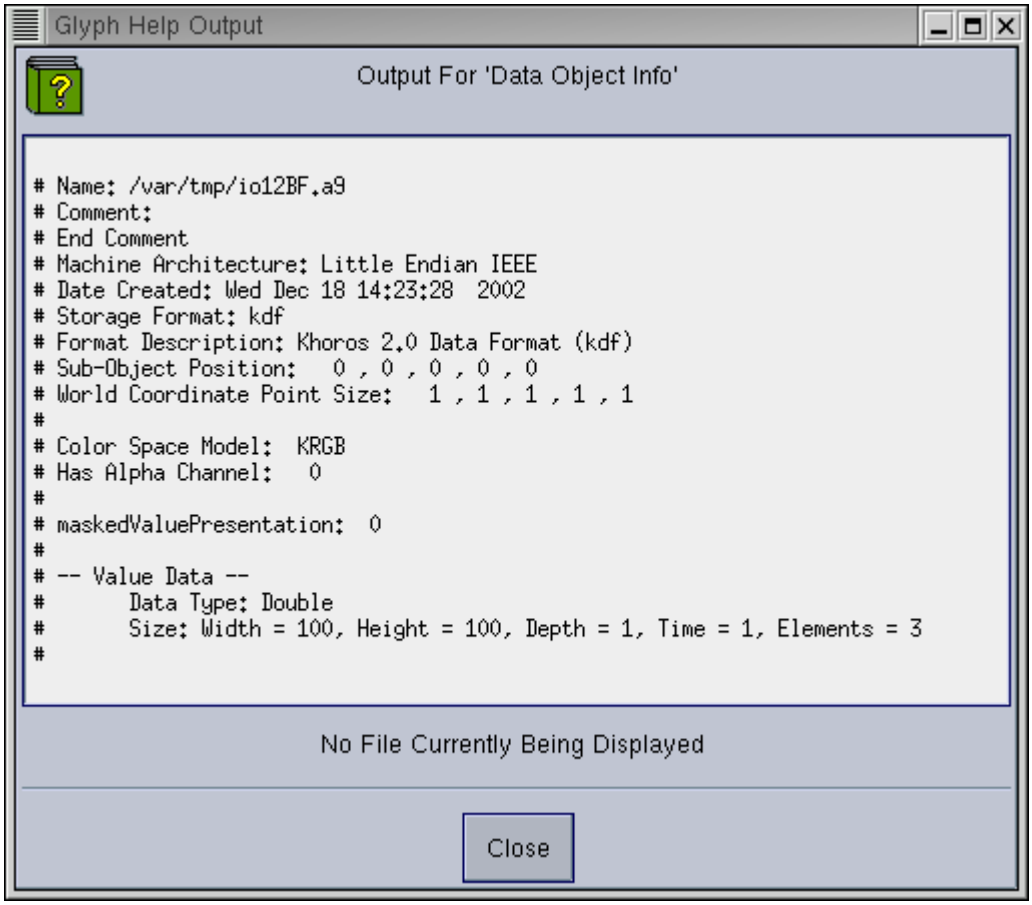
Rezultati izvršavanja programa su prikazani slikama.



Slika 3.66. Početna slika.



Slika 3.67. Slika dobivena kao rezultat programa.



Slika 3.68. Poruka Glypha Data Object Info.

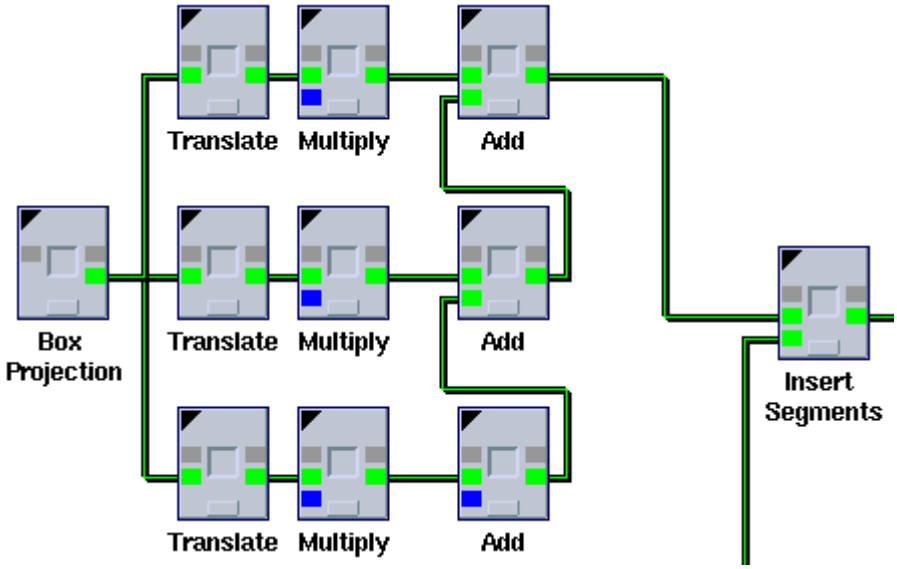
3.4. Hijerarhija, kontrola toka i prevođenje programa.

Isto kao kod procedura tekstualnih programskih jezika, grafičke procedure omogućuju modeliranje programa s obzirom na hijerarhiju. Doprinose boljoj preglednosti grafičkog programa.

3.4.1. Procedure

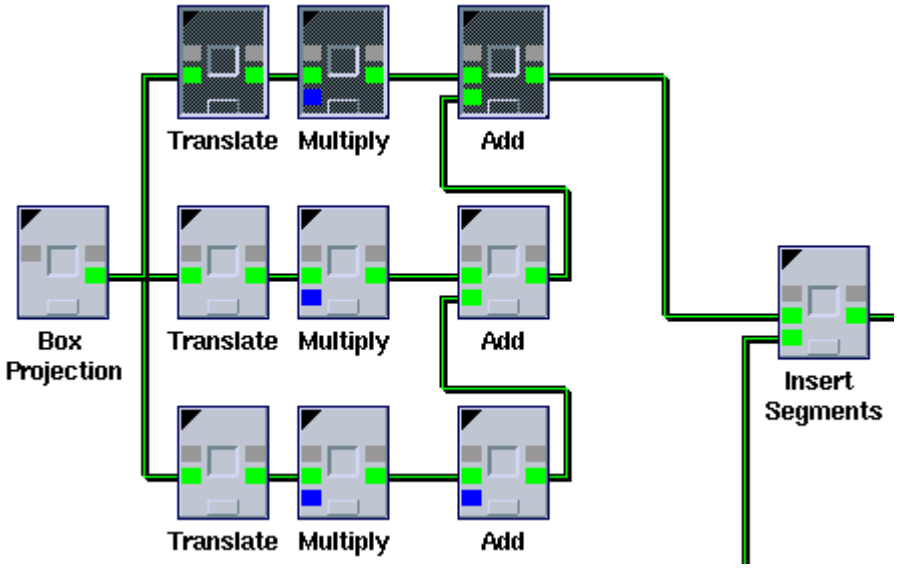
Prikazat će se formiranje procedure koristeći primjer iz prethodnog poglavlja, gdje se malo složeniji program koristi za unos mape u sivu sliku.

Program je prilagođen pretvorbi. Dodan je još jedan Glyph Multiply, kod kojeg je pod parametrima navedena konstanta 1, tako da se ulazna slika množi sa 1, odnosno ne mjenja se. Dodan je još jedan Glyph Add kod kojeg je pod parametrima kao drugi ulaz navedena konstanta 0, tako da se ulazna slika zbraja sa 0, odnosno ne mijenja se. Rezultat izvršavanja programa je isti kao i prije.



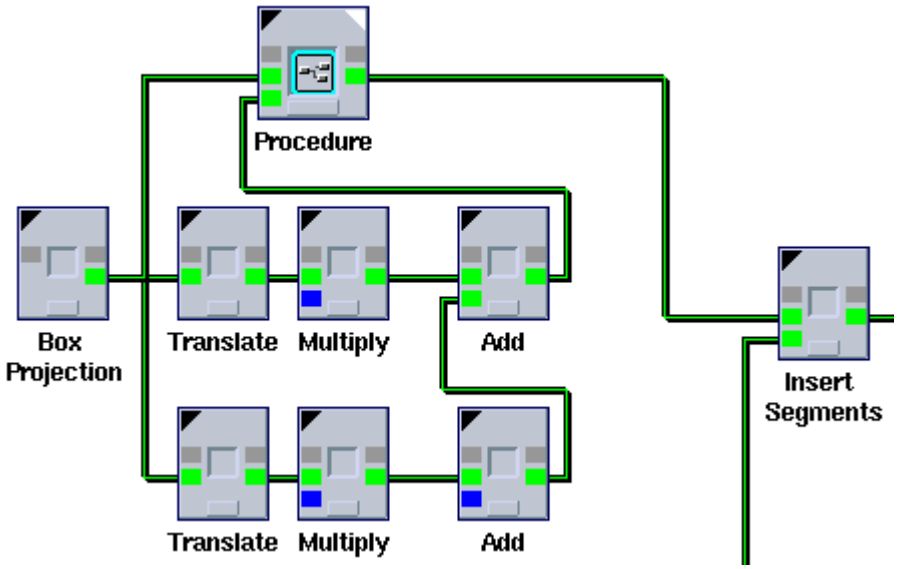
Slika 3.69. Početni izgled programa.

Prvo se označe gornja 3 Glypha, Translate, Multiply i Add, kao što je prikazano slikom 3.70.



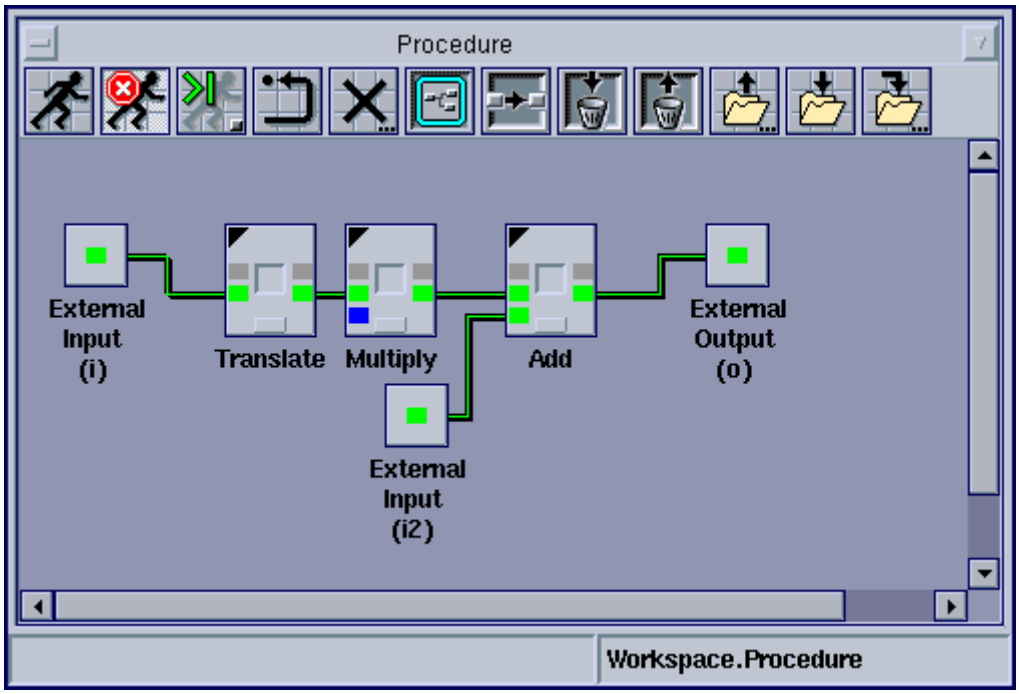
Slika 3.70. Izgled programa nakon označavanja 3 Glypha.

Nakon toga se pokrene kreiranje procedure preko [Control] [Create Procedure]. Glyphovi su zamjenjeni procedurom kao što je prikazano na slici 3.71.



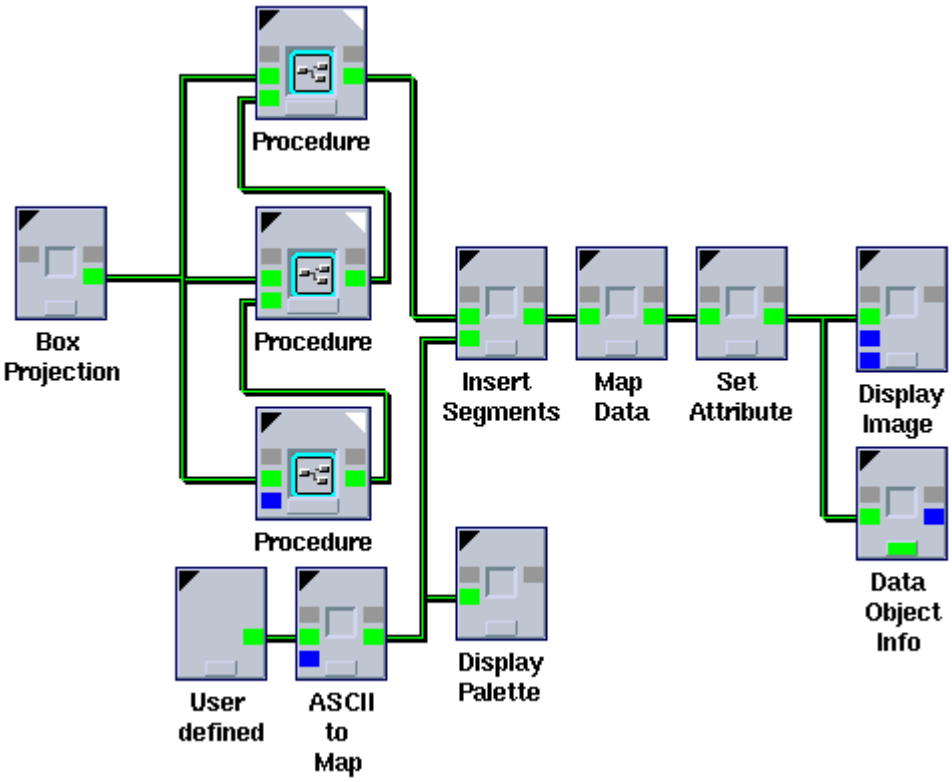
Slika 3.71. Izgled programa nakon formiranja procedure.

Klikom na lijevi crni trokutić Glypha Procedure otvara se parametarski prozor, a klikom na desni bijeli trokutić otvara se prozor sa tijelom procedure, odnosno ulazne i izlazne točke sa Glyphovima od koji se procedura sastoji.



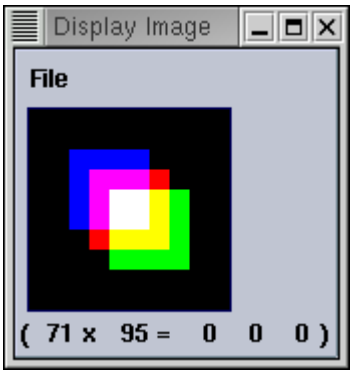
Slika 3.72. Tijelo procedure.

Ponavljajući postupak formiranja procedure zamijene se i ostale dvije kombinacije po 3 Glypha, Translate, Multiply i Add, sa procedurama.



Slika 3.73. Konačni izgled programa sa primjenom procedura.

Nakon unošenja svih parametara u procedure, rezultat izvršavanja programa je isti kao i prije.

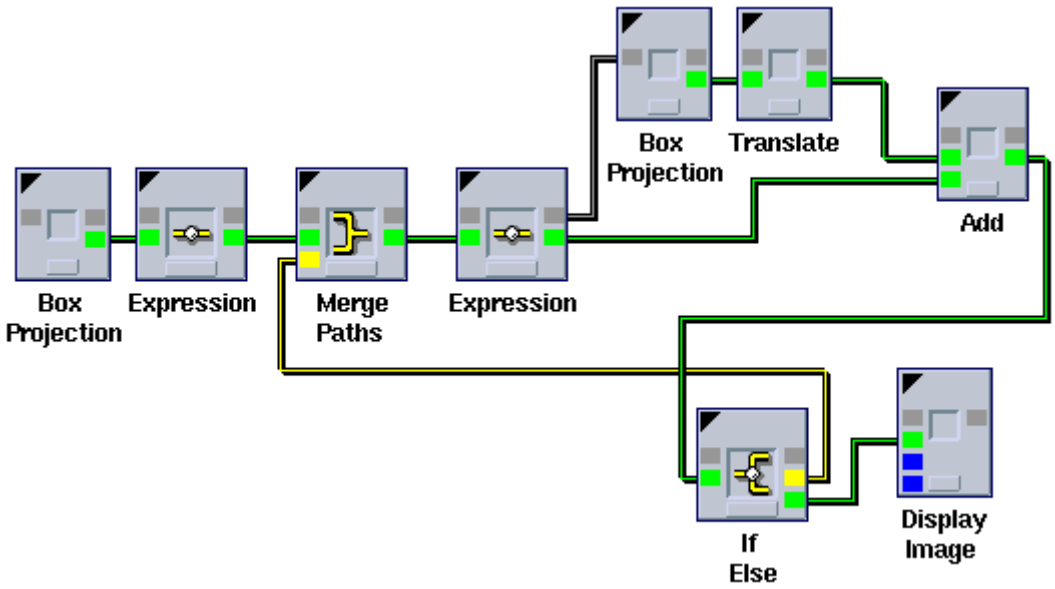


Slika 3.74. Slika dobivena kao rezultat programa.

3.4.2. Petlje

Slično kao u primjeru sa nizom pravokutnika različitih sivih razima, formirat će se niz pomoću petlje čija će kontrolna varijabla određivati sivu nijansu i iznos translacije.

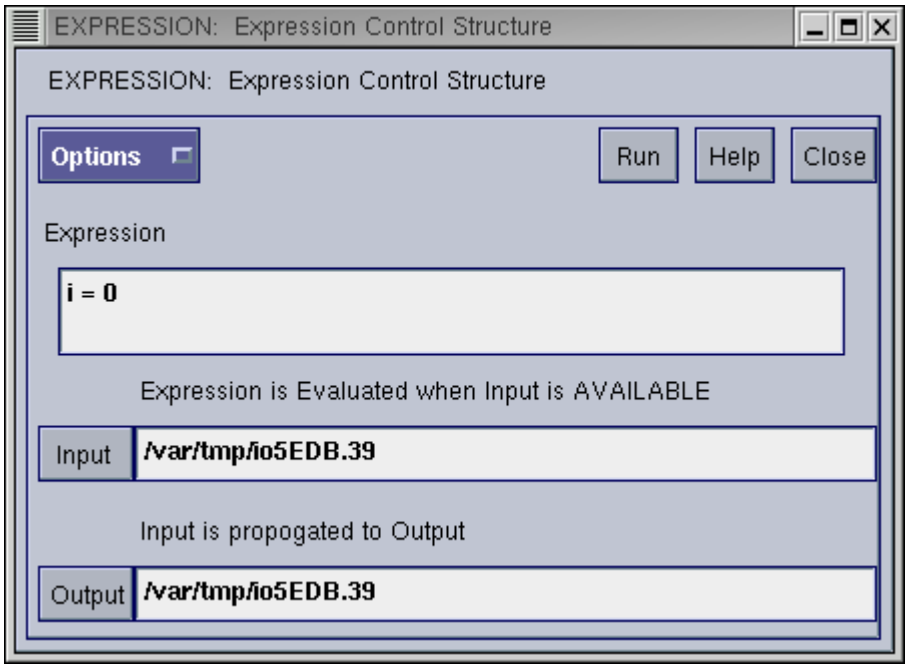
Na radnu površinu potrebno je postaviti i povezati Glyphove:
[Glyphs] [Input/Output] [Conversion] [Box Projection],
[Glyphs] [Program Utilities] [Conditional Construct] [Expression],
[Glyphs] [Program Utilities] [Conditional Construct] [Merge Paths],
[Glyphs] [Data Manip] [Reorganize Data] [Translate],
[Glyphs] [Arithmetic] [Two Operand Arithmetic] [Add],
[Glyphs] [Program Utilities] [Conditional Construct] [If Else],
[Glyphs] [Visualization] [Non-Interactive Image Display] [Display Image].



Slika 3.75. Primjer petlje u programu.

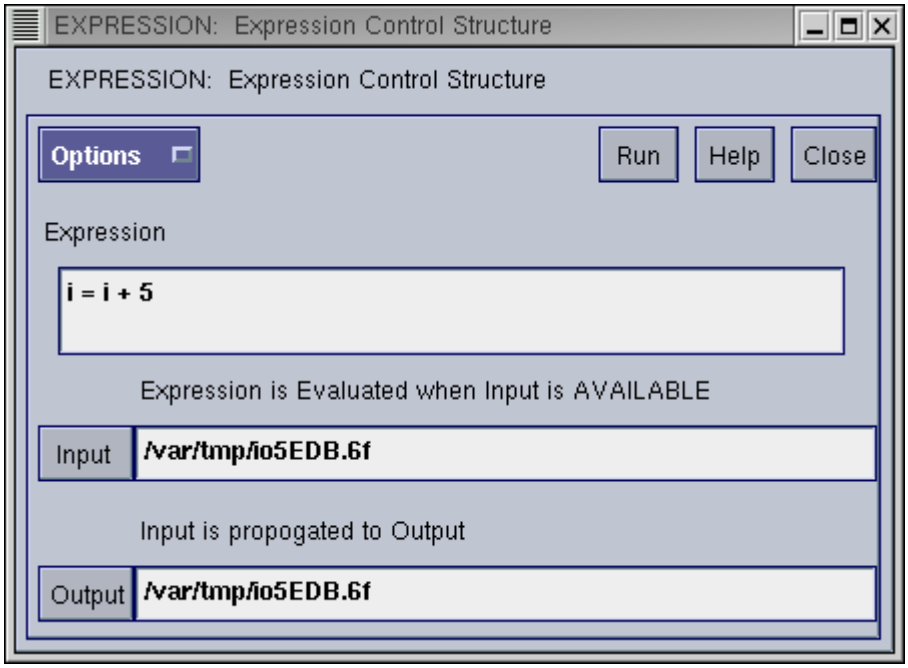
Parametri Glypha Box Projecton postave se kao u prethodnim primjerima. Prvi Glyph Box Projecton generira pravokutnik samo jedanput, dok drugi generira pravokutnik inteziteta određenog varijablom, svaki put nakon izvršenja Glypha Expression.

Glyph Expression predaje ulaz izlazu, uz izvršavanje određenog računskog izraza. Prvi Glyph Expression inicijalizira kontrolnu varijablu petlje.



Slika 3.76. Parametri za prvi Expression Glyph.

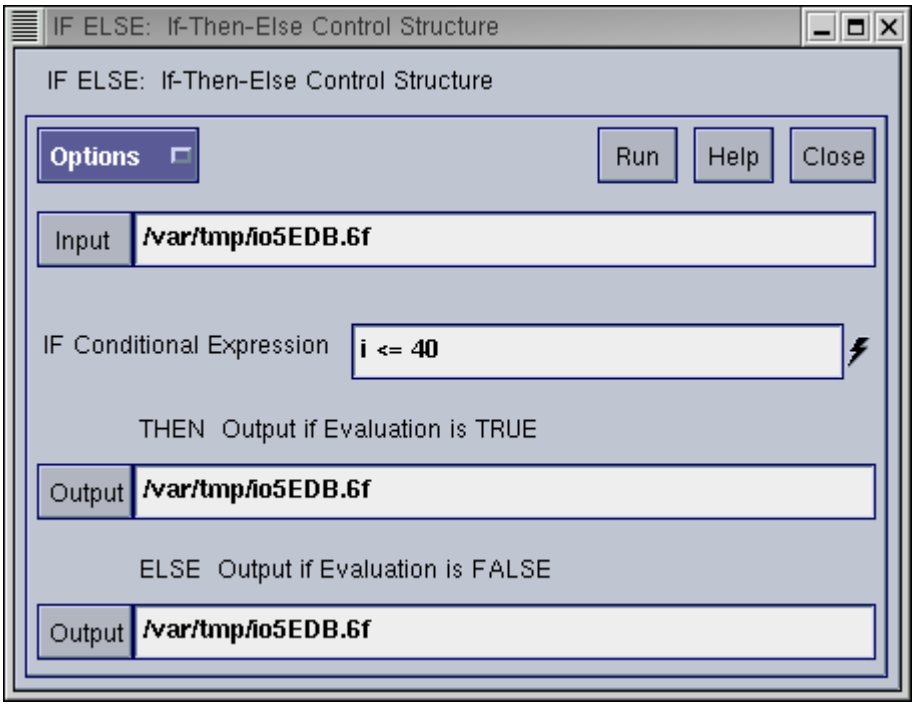
Početna vrijednost kontrolne varijable je 0. Drugi Glyph Expression uvećava kontrolnu varijablu za zadani korak.



Slika 3.77. Parametri za drugi Expression Glyph.

Glyph Merge Paths propušta sa ulaza na izlaz onu ulaznu vrijednost koja se brže mijenja. To znači da će se jednom propustiti inicijalizacija varijable, nakon čega se svaki put propušta izlaz If Else Glypha.

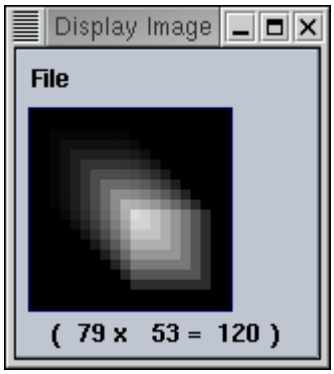
Glyph Add zbraja staru sliku sa novim pravokutnikom inteziteta određenog kontrolnom varijablom i translatiranog za iznos kontrolne varijable. Rezultat dolazi do Glypha If Else, gdje se ispituje da li je kontrolna varijabla postigla željenu vrijednost.



Slika 3.78. Parametarski prozor za Glyph If Else.

Dok kontrolna varijabla ne postigne željeni iznos, slika se šalje preko gornjeg izlaza Glypha If Else na ulaz Glypha Merge Paths i dalje u tijelo petlje uz povećavanje vrijednosti kontolne varijable.

Kad kontrolna varijabla postigne željeni iznos, slika se šalje preko donjeg izlaza Glypha If Else i prikazuje.

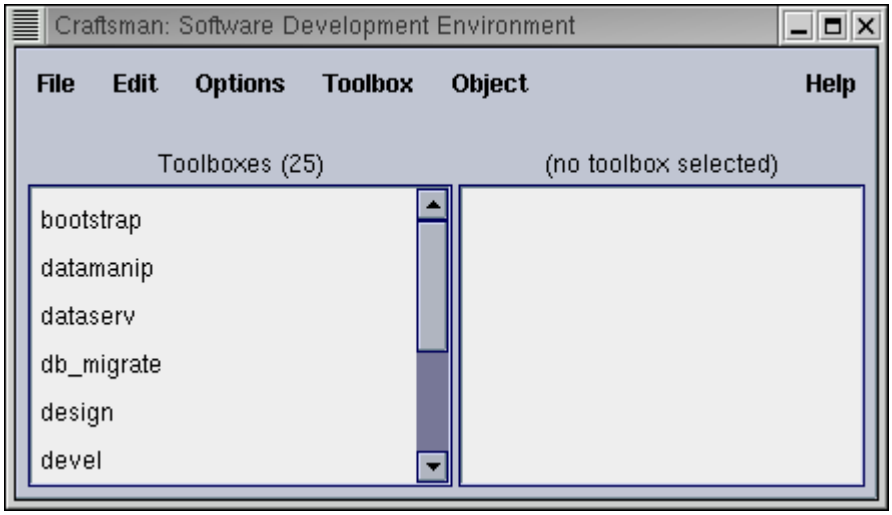


Slika 3.79. Rezultat izvođenja programa.

3.4.3. Kreiranje Toolboxa

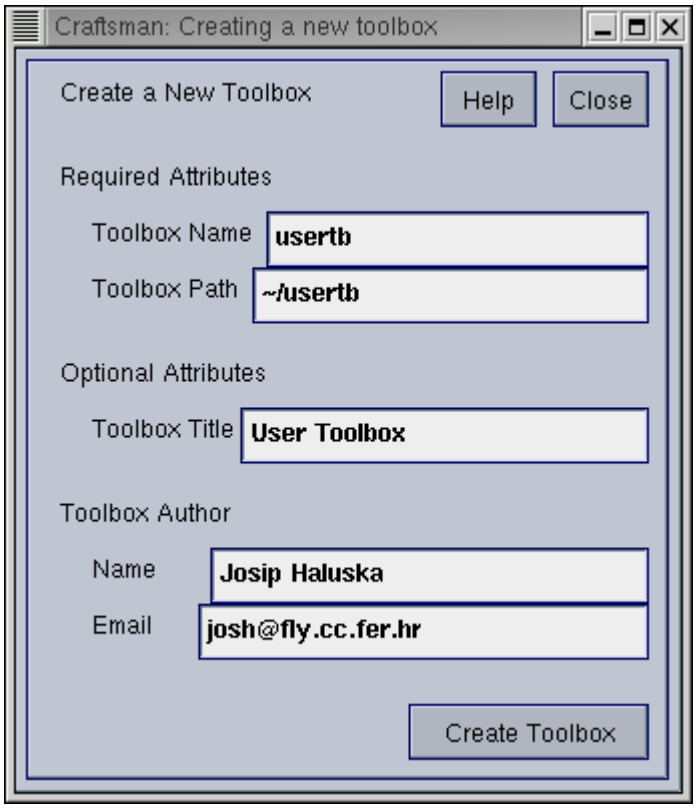
Svaki program Khorosa nalazi se u Toolboxu. Toolbox je skup programa i biblioteka koji čine entitet, odnosno objekt. Da bi se grafički program načinjen pomoću Cantate preveo u operator koji se kasnije može pozivati iz komandne linije kao samostalna aplikacija, potrebno je prvo kreirati Toolbox. Pokretanjem Craftsmana pojavljuje se prozor sa popisom postojećih Toolboxa.

```
%craftsman &
```



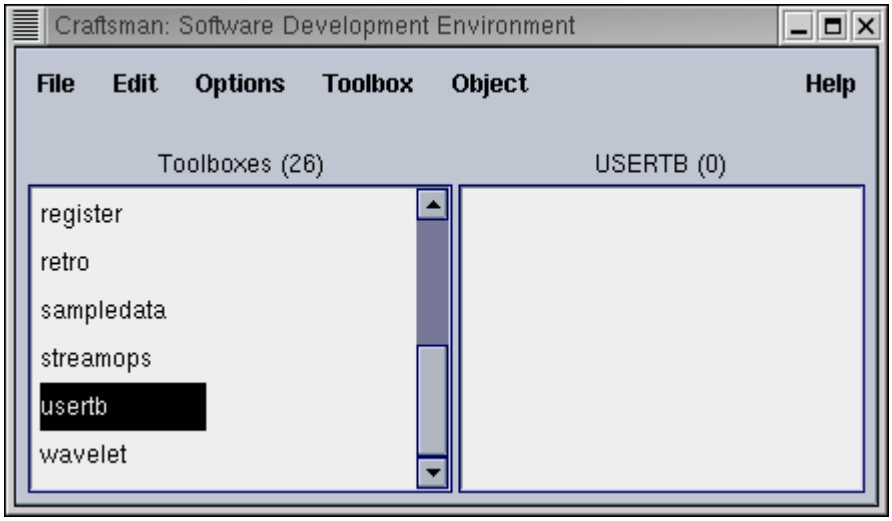
Slika 3.80. Craftsman sa popisom Toolboxa.

Za kreiranje novog Toolboxa potrebno je odabrati [Toolbox] [Create Toolbox...] te unijeti tražene podatke.



Slika 3.81. Podaci potrebni za kreiranje Toolboxa.

Klikom [Create Toolbox] generira se novi Toolbox.



Slika 3.82. Craftsman sa popisom Toolboxa.

3.4.4. Prevođenje grafičkog programa

U čemu je prednost prevedenog programa sastavljenog od mreže Glyphova, kad se i pomoću procedura može program dobro hijerarhijski organizirati. Postoji nekoliko razloga za prevođenje programa.

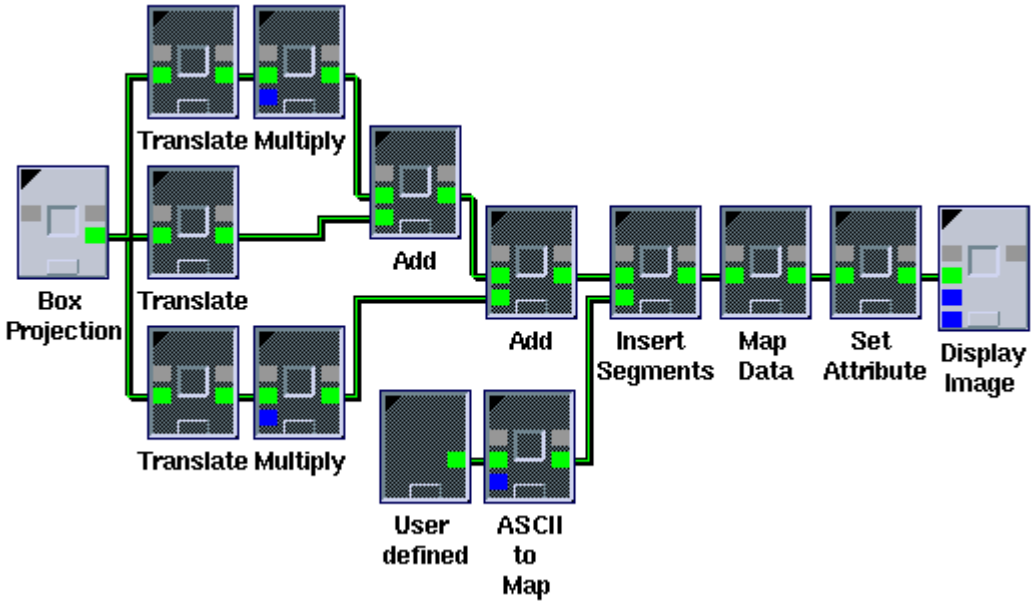
Prevedeni program se predstavlja novim Glyphom, do kojeg se može doći preko Glyphs izbornika Cantate. On se može koristiti u drugim programima kao i svaki standardni Glyph.

Prevedeni program predstavlja operator koji je neovisan o Cantati i može se pokretati iz komandne linilje kao i drugi standardni operatori.

Problem kod programiranja pod grafičkim sučeljem je sporo izvršavanje grafičkih programa. Što više raste složenost programa, primjećuje se i pad brzine izvršavanja. Prevedeni program je znatno brži.

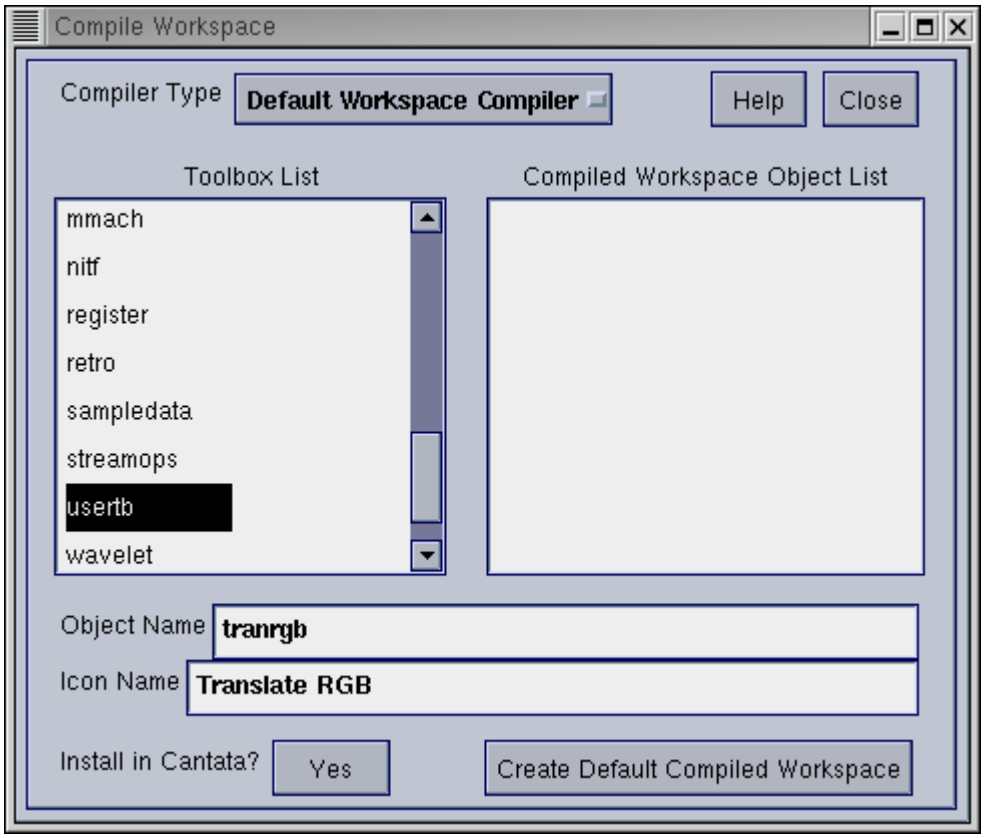
Nema problema kod prijenosa prevedenog programa na druga računala, jer je on neovisan o sustavu Khoros.

Kao program koji će se prevesti, poslužit će raniji primjer ubacivanja palete boja u sliku pravokutnika sa sivim tonovima. Slično kao i kod generiranja procedura, prvo se označe Glyphovi koji će činiti prevedeni program.



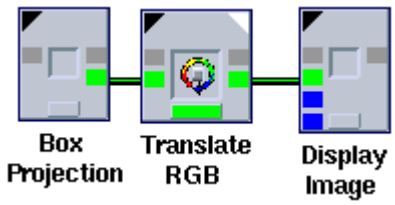
Slika 3.83. Izgled programa nakon označavanja Glyphova.

Nakon odabira [Wokspace] [Compile Workspace] popunjavju se parametri prozora. Odabire se Toolbox generiran u prijašnjem primjeru.



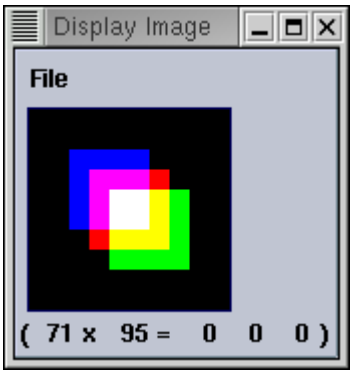
Slika 3.84. Navođenje parametara za prevođenje programa.

Klikom na [Create Default Compiled Workspace] prevodi se program sastavljen od označenih Glyphova, koji se zamjenjuju dobivenim Glyphom.



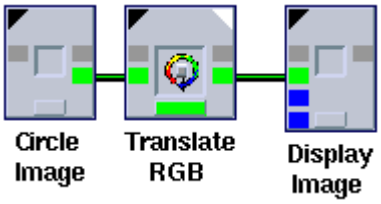
Slika 3.85. Prevedeni program prikazan Glyphom.

Rezultat izvršavanja programa prikazan je slikom 3.86.



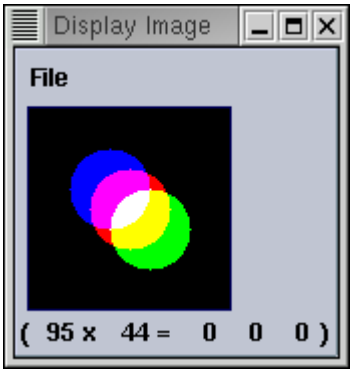
Slika 3.86. Slika dobivena kao rezultat programa.

Umjesto pravokutnika kao ulaz programa može se koristiti i krug, odabirom [Glyphs] [Input/Output] [Conversion] [Circle Image].



Slika 3.87. Prevedeni program prikazan Glyphom.

Rezultat izvršavanja programa prikazan je slikom 3.88.



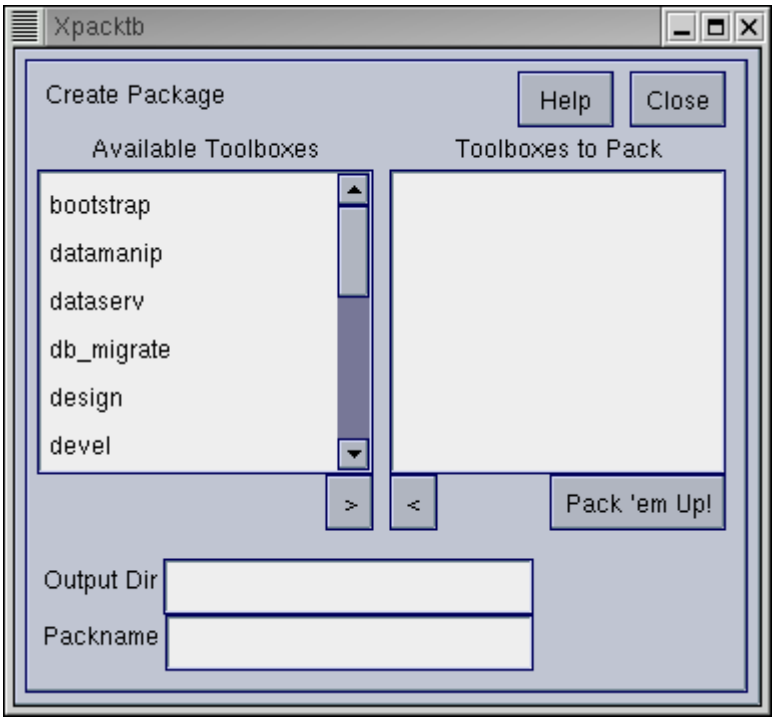
Slika 3.88. Slika dobivena kao rezultat programa.

3.4.5. Priprema prevedenog programa za prijenos na druga računala.

Prevedeni program je potpuno neovisan o Cantati, ali pri izvođenju iz komandne linije poziva operatore Glyphova od kojih se prevedeni program sastoji. Da bi bilo moguće izvođenje programa na nekom drugom računalu, potrebno je uz binarnu datoteku prevedenog programa, na drugo računalo prenijeti i binarne datoteke svih Glyphova koje prevedeni program sadrži.

Za pakiranje Toolboxa može se koristiti alat kpacktb, ili xpacktb koji je identičan kpacktb, ali za odabir parametara koristi grafičko sučelje.

```
%xpacktb &
```



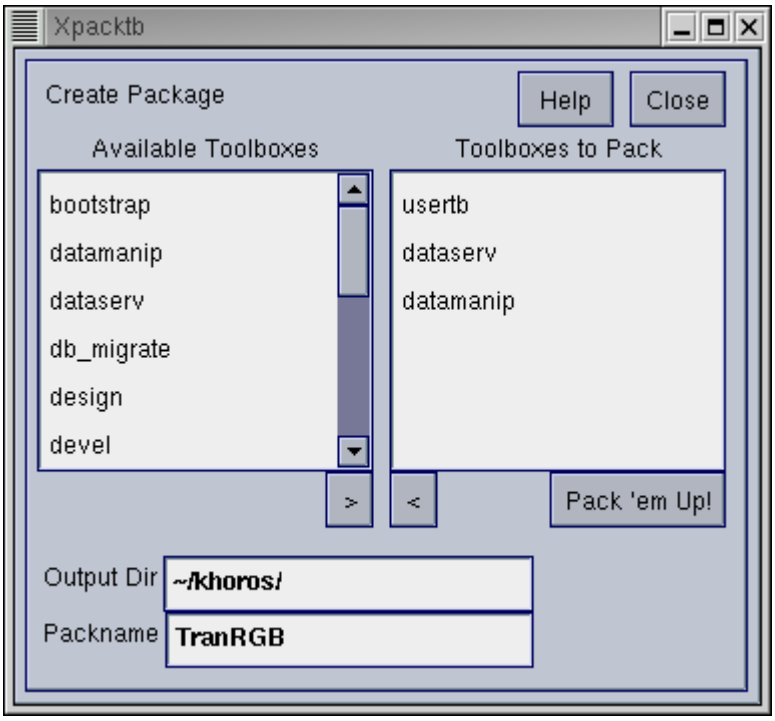
Slika 3.89. Poziv programa xpacktb.

Nakon pokretanja, potrebno je odabrati Toolbox u kojem se nalazi prevedeni program i sve one u kojima se nalaze korišteni Glyphovi.

Za vrijeme izvođenja programa, u tekstualnom prozoru Cantate prikazuju se ekvivalentne naredbe komandne linije. Prikazan je primjer izvođenja prevedenog programa.

```
# (new) USERTB tranrgb Translate RGB
$USERTBBIN/tranrgb -i "/var/tmp/io716C.0" -o "/var/tmp/io716C.20"
```

Prva linija sadrži ime Toolboxa i Glypha, a druga linija ekvivalentnu naredbu sa potrebnim parametrima. Praćenjem sadržaja tekstualnog prozora određuju se i Toolboxi za ostale Glyphove. Zajedno sa potrebnim Glyphovima prenose se i svi oni od kojih se pojedini Toolbox sastoji.



Slika 3.90. Odabrani su pojedini Toolboxi.

Klikom na [Pack'em Up!] obavlja se pakiranje programa za prijenos na druga računala. Proces pakiranja koristi standardne Unix programe za arhiviranje, tar i gzip, a kao rezultat dobiva se datoteka "TranRGB.kpg".

Uz dobivenu datoteku "TranRGB.kpg", potrebno je na ciljno računalo prenijeti i skript datoteku kpkgadd, koja služi za instalaciju dobivene datoteke.

Nakon što se na ciljno računalo prenesu obje datoteke, te na njemu postoje programi gzip i tar, može se napraviti instalacija prevedenog programa.

```
%kpkgadd TranRGB.kpg
```

4. Riješavanje problema korištenjem Cantate

4.1. Registarske tablice vozila

U primjeru koji slijedi pokazat će se način određivanja položaja registarske tablice vozila i pokušaj izdvajanja sadržaja tablice. Za rješenje problema koristi će se Cantata.

4.1.1. Način riješavanja

Kao ulaz u program koristit će se RGB slike stražnjih strana automobila. Slike su dimenzija 640x480 i pohranjene su u JPEG formatu.

Ulaznu sliku potrebno je pretvoriti u sliku sa sivim tonovima. Pretvorba se obavlja izdvajanjem pojednih RGB komponenti, njihovim zbrajanjem, te dijeljenjem dobivenog zbroja sa 3.

Dobivenu sivu sliku potrebno je poboljšati. Za poboljšanje sive slike koristi će se metode izjednačavanja histograma i rastezanja kontrasta.

Oba postupka se ubrajaju u metode operacije na točki, sa zajedničkom karakteristikom da izlazna vrijednost sive razine točke ovisi samo o ulaznoj vrijednosti sive razine u istoj točki.

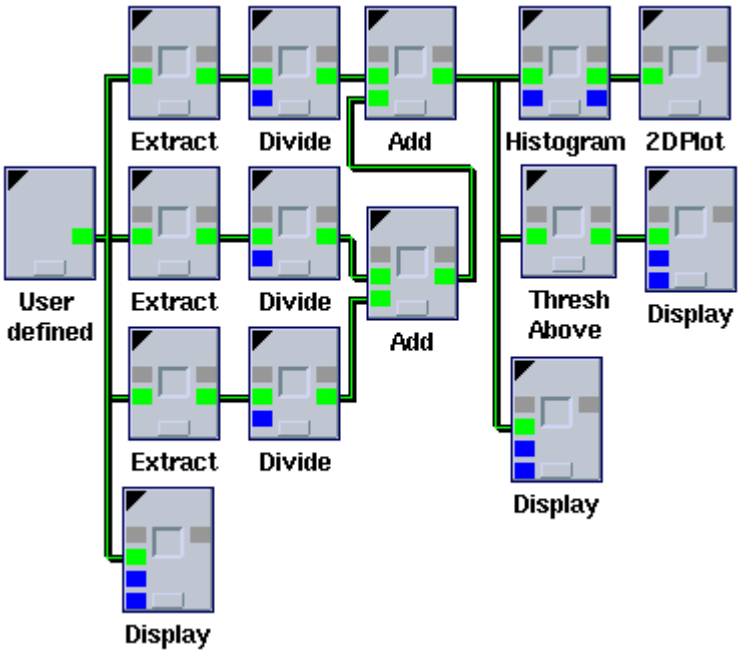
Histogram slike predstavlja relativnu frekvenciju pojave različitih sivih razina u slici. Tehnike modeliranja histograma mijenjaju sliku, tako da se dobije histogram željenog oblika. Izjednačavanje histograma je djelotvorna metoda za poboljšanje slike, a kao rezultat dobije se jednolika raspodjela sivih tonova.

Svrha postupka rastezanja kontrasta je dobivanje slike sa povećanim kontrastom. Niski kontrast može biti rezultat nejednolikog osvjetljenja scene ili slabog dinamičkog opsega senzora, pa je mala razlika između svijetlih i tamnih dijelova slike što otežava interpretaciju sadžaja slike. Za dobivanje bolje slike potrebno je uski pojas sivih tonova razvući na širi interval.

Nakon poboljšanja, na sliku se primjenjuje Thresholding, sa pragom određenim na osnovu lokalnih minimuma histograma. Kao rezultat dobiva se binarna slika u kojoj bi trebala biti uočljiva registarska tablica vozila.

4.1.2. Program za riješavanje problema

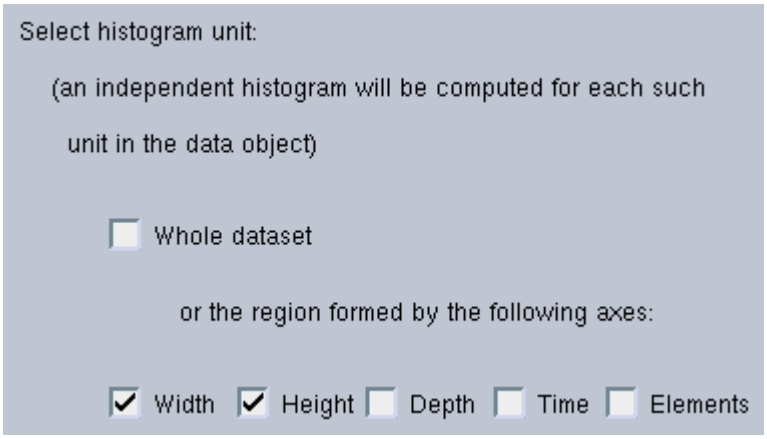
Prikazan je program za riješavanje problema.



Slika 4.1. Program za uočavanje registarske tablice.

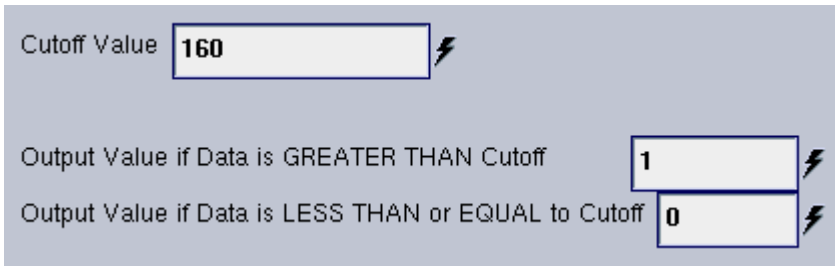
Glyph Extract izdvaja pojedinu RGB komponentu. Izdvojena komponenta se pomoću Glypha Divide dijeli sa 3. Pomoću Glyphova Add zbrajaju se rezultati dijeljenja i kao izlaz dobije se siva slika.

Glyph Histogram preko Glypha 2D Plot prikazuje histogram dobivene sive slike. Osim histograma slike mogu se prikazati histogrami preko pojedinih dimenzija, visine ili širine slike, što se određuje u parametarskom prozoru Glypha.



Slika 4.2. Dio parametarskog prozora Glypha Histogram.

Glyph Thresh Above generira binarnu sliku, na osnovu zadanog praga.



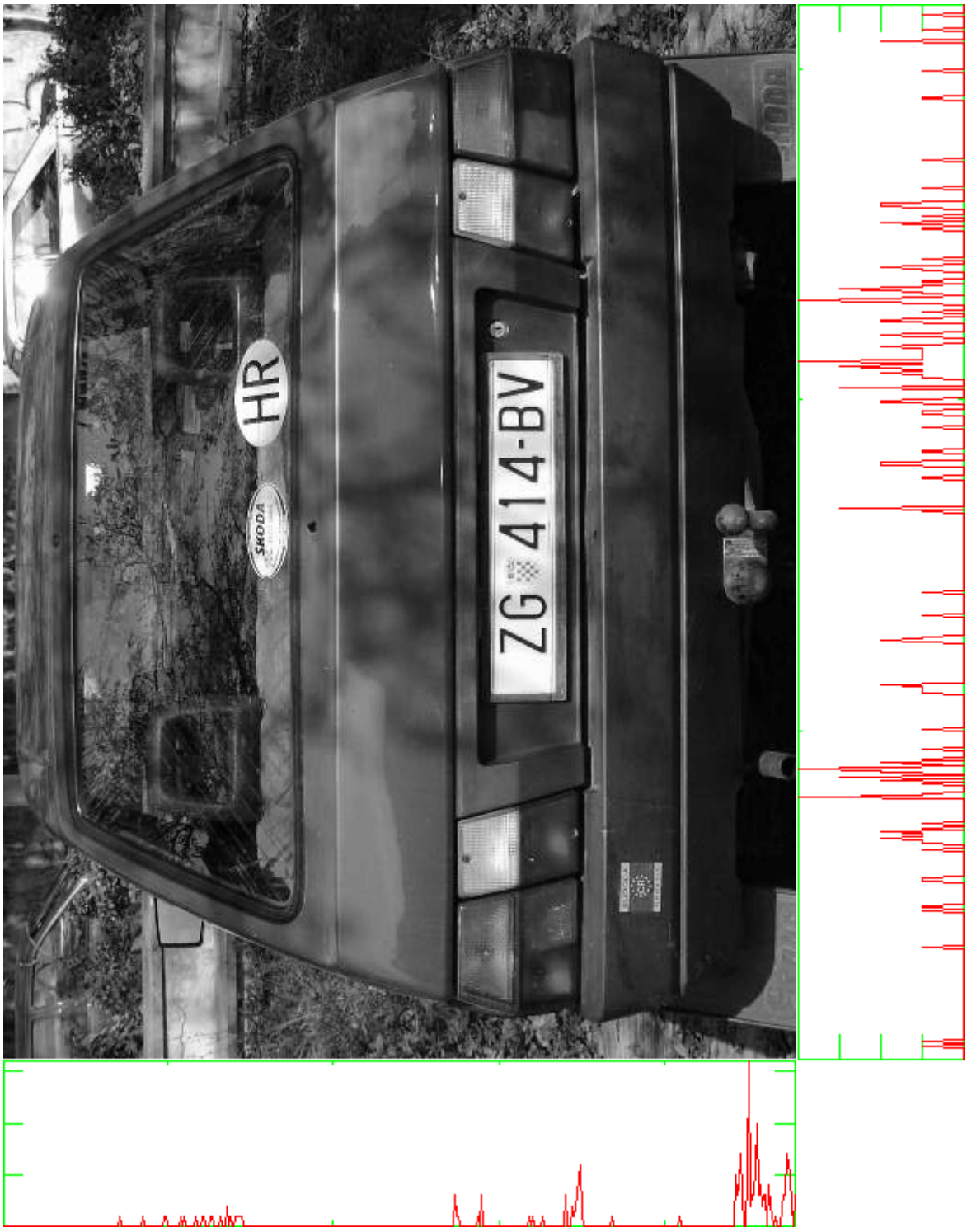
Slika 4.3. Dio parametarskog prozora Glypha Thresh Above.

4.1.3. Određivanje položaja tablice

Na ulazu u program nalazi se prikazana RGB slika.

Slika 4.4. Ulazna RGB slika.

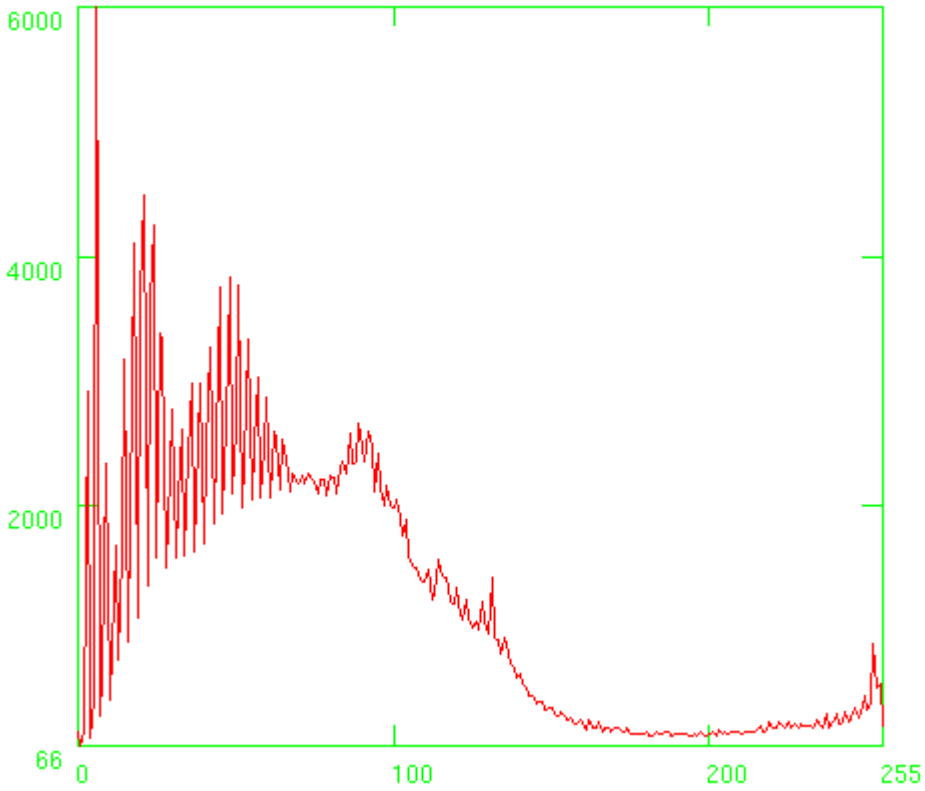
Položaj tablice može se odrediti pomoću histograma dobivene sive slike, preko dimenzije visine i širine. Obično se nagli prijelazi sivih razina mogu vezati uz skokove vrijednosti histograma, pa se na osnovu približnih dimenzija tablice i približnog mjesta gdje bi se na slici ona mogla pojaviti, može odrediti njezin položaj.



Slika 4.5. Histogrami dobivene sive slike po dimenziji širine i visine.

4.1.4. Generiranje binarne slike

Na osnovu histograma sive slike, pokušava se odrediti lokalni minimum, odnosno prag za koji će Threshold dati najbolji rezultat.



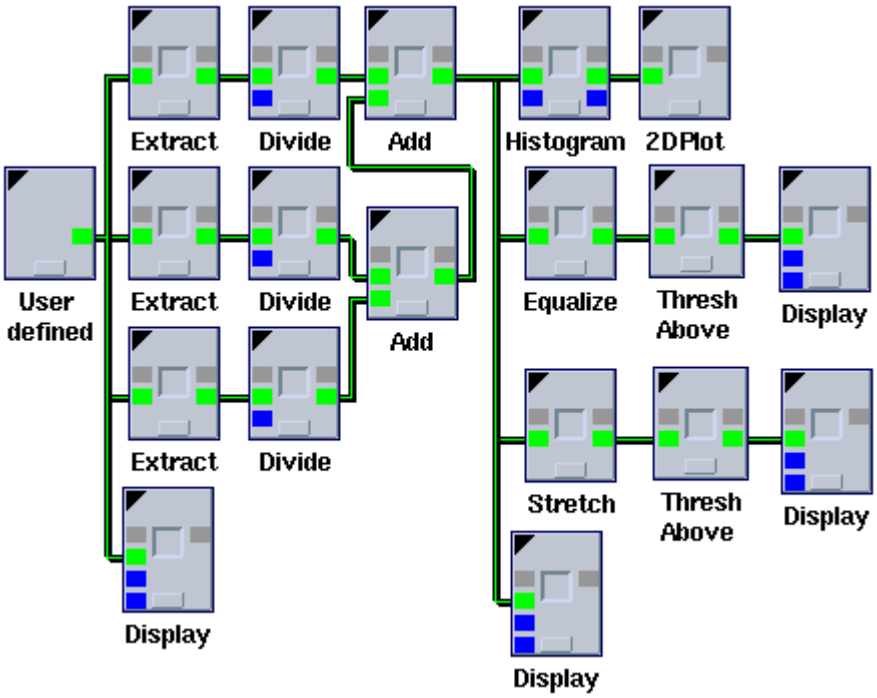
Slika 4.6. Histogram dobivene sive slike.

Na osnovu histograma za prag se uzima vrijednost 200. Nakon određivanja praga, kao rezultat programa dobije se binarna slika sa uočljivom tablicom vozila.

Slika 4.7. Rezultat izvođenja programa.

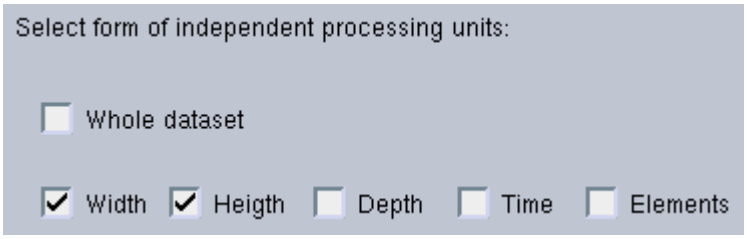
4.1.5. Uvođenje postupaka poboljšanja sive slike

Za poboljšanje sive slike koristit će se metode izjednačavanja histograma i rastezanja kontrasta. Programu je potrebno dodati još nekoliko Glyphova tako da izgleda kao program prikazan na slici.

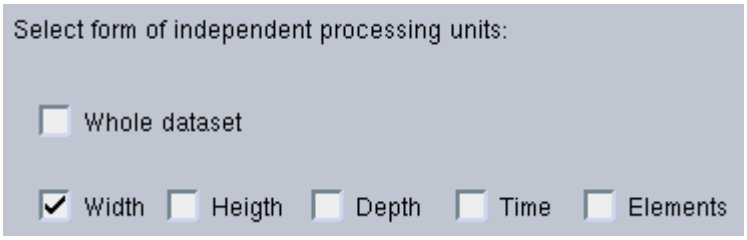


Slika 4.8. Program za uočavanje tablice uz poboljšavanje sive slike.

Glyph Equalize obavlja izjednačavanje histograma, a Glyph Stretch rastezanje kontrasta. Kod oba Glypha se mogu odrediti dimenzije preko kojih će se operacija izvesti.



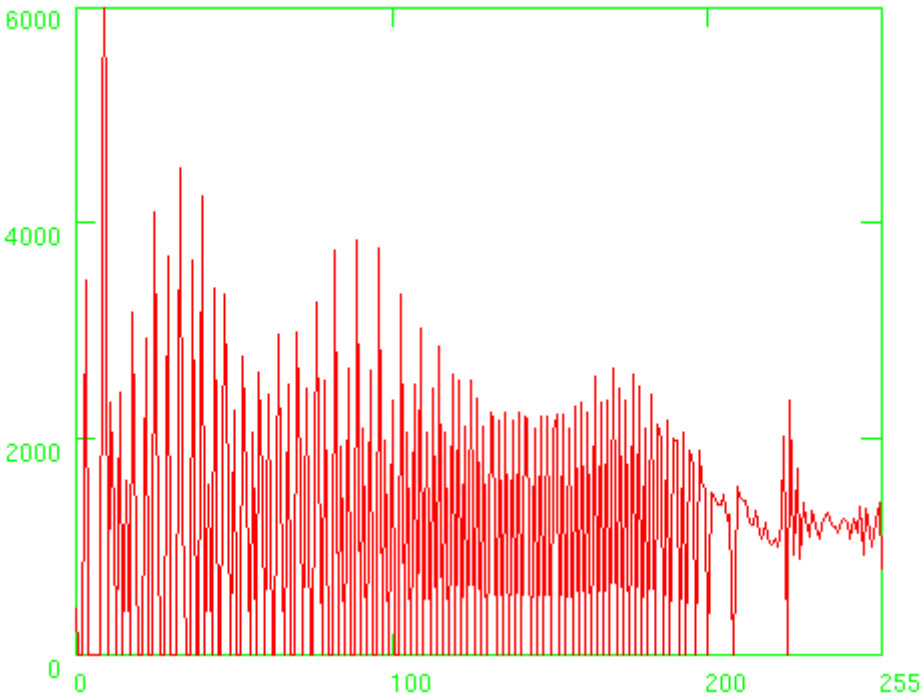
Slika 4.9. Dio parametarskog prozora Glypha Equalize.



Slika 4.10. Dio parametarskog prozora Glypha Stretch.

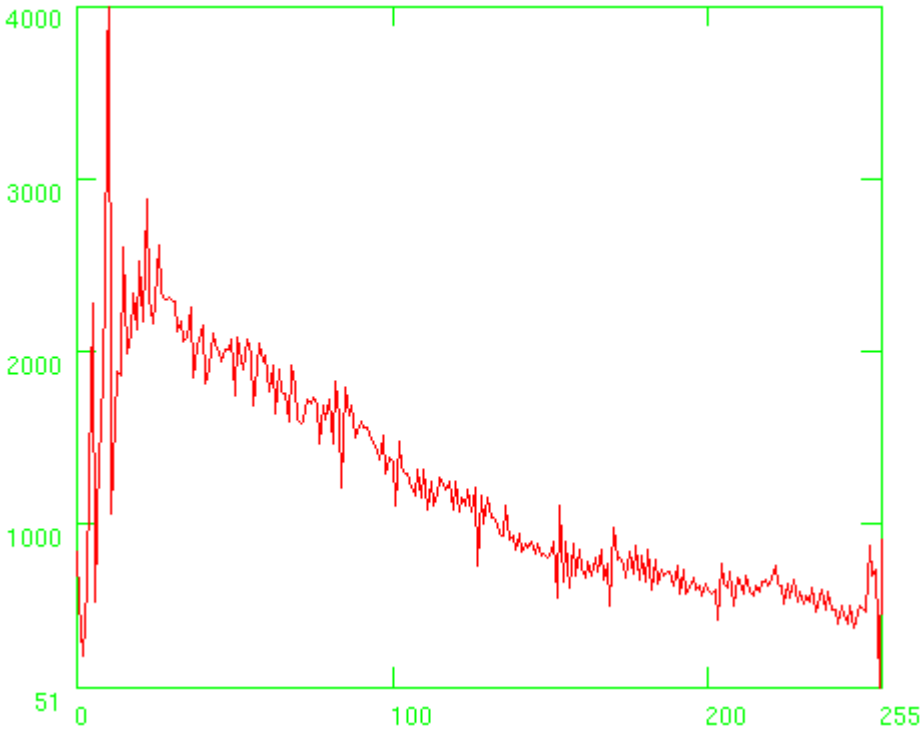
Kao rezultat izvođenja programa prikazuju se poboljšane sive slike i njihovi histogrami.

Slika 4.11. Slika dobivena metodom izjednačavanja histograma.



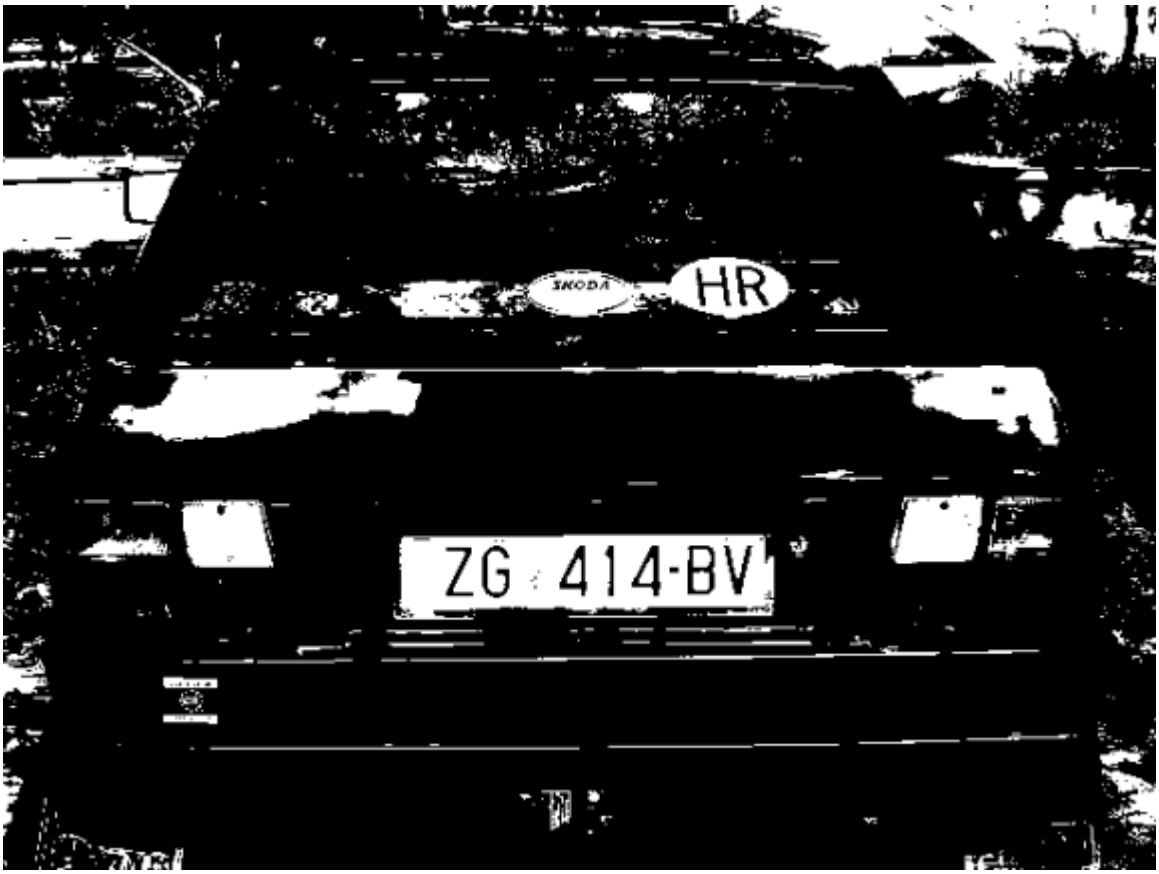
Slika 4.12. Histogram slike dobivene metodom izjednačavanja histograma.

Slika 4.13. Slika dobivena metodom rastezanja kontrasta.



Slika 4.14. Histogram slike dobivene metodom rastezanja kontrasta.

Na osnovu histograma, za prag kod slike sa izjednačavanjem histograma uzima se vrijednost 220, a za prag kod slike sa rastezanjem kontrasta uzima se vrijednost 200. Nakon određivanja praga, kao rezultat programa dobiju se binarne slike.



Slika 4.15. Binarna slika dobivena metodom izjednačavanja histograma.

Slika 4.16. Binarna slika dobivena metodom rastezanjem kontrasta.

Na osnovu primjera poboljšanja slike, vidi se da poboljšanje slike korištenjem navedenih metoda nema velikog utjecaja na krajnji rezultat. Pragovi dobiveni na osnovu poboljšanih histograma, ne razlikuju se značajno od praga koji je određen prema histogramu početne sive slike.

Kako bi se vidjeli rezultati programa za ne tako idealnu sliku kao u prijašnjem primjeru, pokušat će se dobiti što bolji rezultati za sliku automobila snimljenu po mraku.

Slika 4.17. Ulazna RGB slika.

Prikazana je dobivena siva slika i njezin histogram.

Slika 4.18. Siva slika.

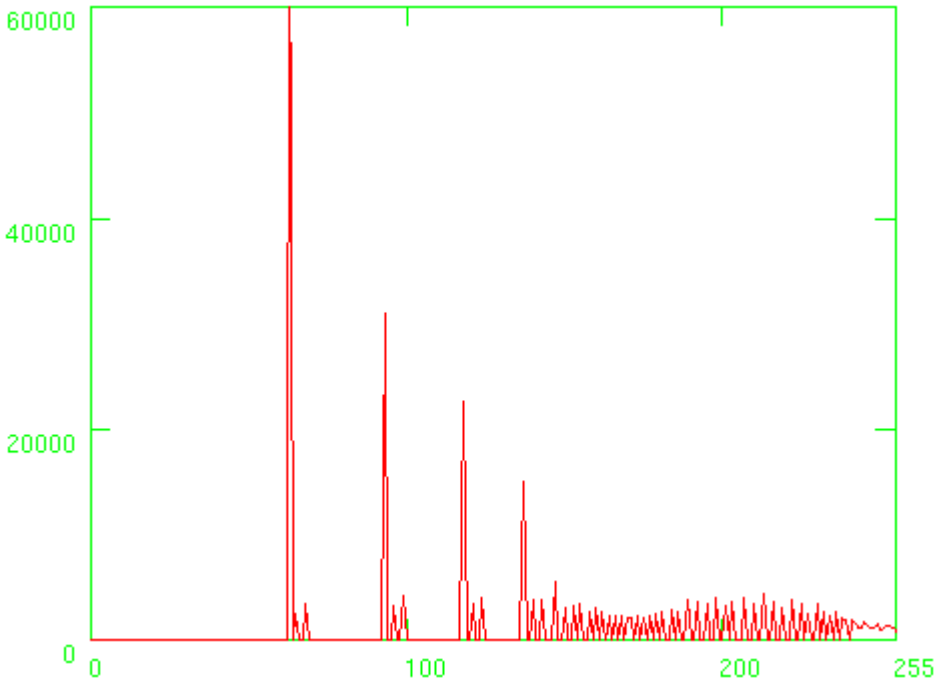
Slika 4.19. Histogram sive slike.

Na osnovu histograma, za prag se uzima vrijednost 40. Nakon određivanja praga, kao rezultat programa dobije se binarna slika.

Slika 4.20. Rezultat izvođenja programa.

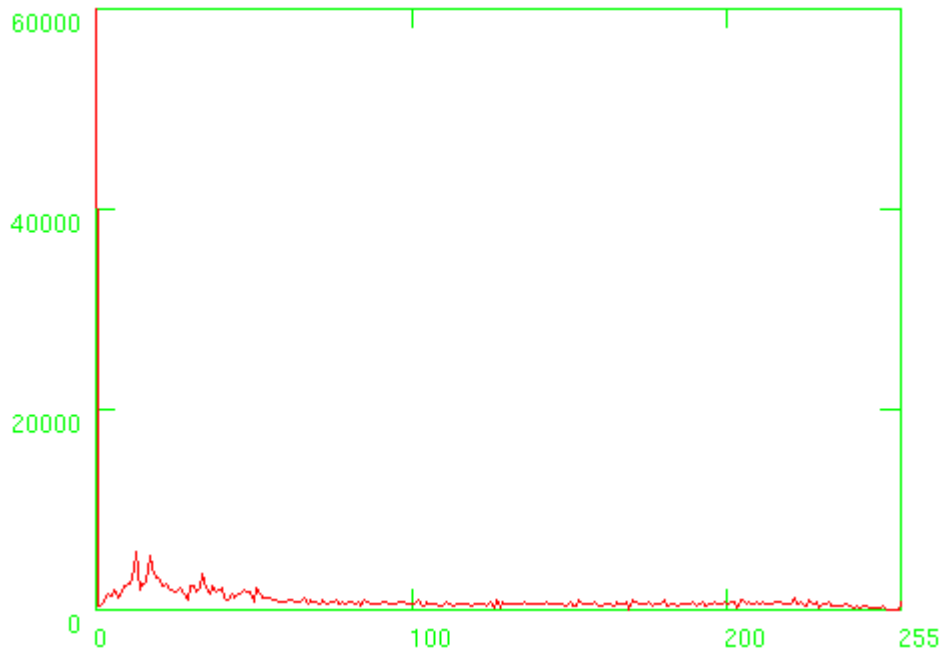
Slika je snimljena po mraku, pa bi metode poboljšanja slike trebale imati značajni utjecaj na krajnje rezultate. Kao rezultat izvođenja programa prikazuju se poboljšane sive slike i njihovi histogrami.

Slika 4.21. Slika dobivena metodom izjednačavanja histograma.



Slika 4.22. Histogram slike dobivene metodom izjednačavanja histograma.

Slika 4.23. Slika dobivena metodom rastezanja kontrasta.



Slika 4.24. Histogram slike dobivene metodom rastezanja kontrasta.

Na osnovu histograma, za prag kod slike sa izjednačavanjem histograma uzima se vrijednost 220, a za prag kod slike sa rastezanjem kontrasta uzima se vrijednost 180. Nakon određivanja praga, kao rezultat programa dobiju se binarne slike.

Slika 4.25. Binarna slika dobivena metodom izjednačavanja histograma.

Slika 4.26. Binarna slika dobivena metodom rastezanja kontrasta.

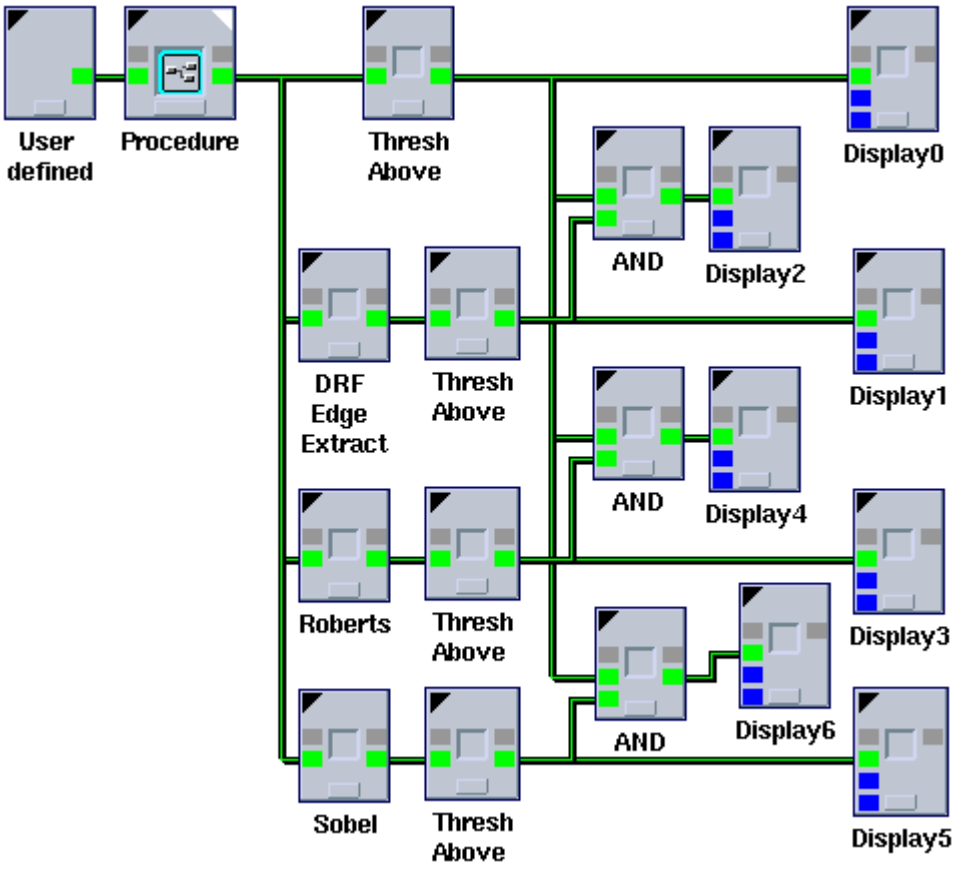
Na osnovu primjera poboljšanja slike, vidi se da siva slika dobivena pomoću metoda poboljšanja izgleda bolje, ali korištenje navedenih metoda nema velikog utjecaja na krajnji rezultat.

U konkretnom primjeru metoda rastezanja kontrasta daje slabije rezultate nego metoda izjednačavanja histograma.

Prag slike bez korištenja metoda poboljšanja postavljen je na 40, a korištenjem metoda na približno 200, pa se može zaključiti da je dobra strana metoda poboljšanja grupiranje vrijednosti pragova oko 200 bez obzira na sliku.

4.1.6. Preklapanje binarnih slika sa detektiranim rubovima

Na primjerima koji slijede vidjet će se rezultati preklapanja dobivene binarne slike sa slikama dobivenim nekom od metoda za detekciju rubova. Dio programa za dobivanje sive slike zamjenjen je procedurom.



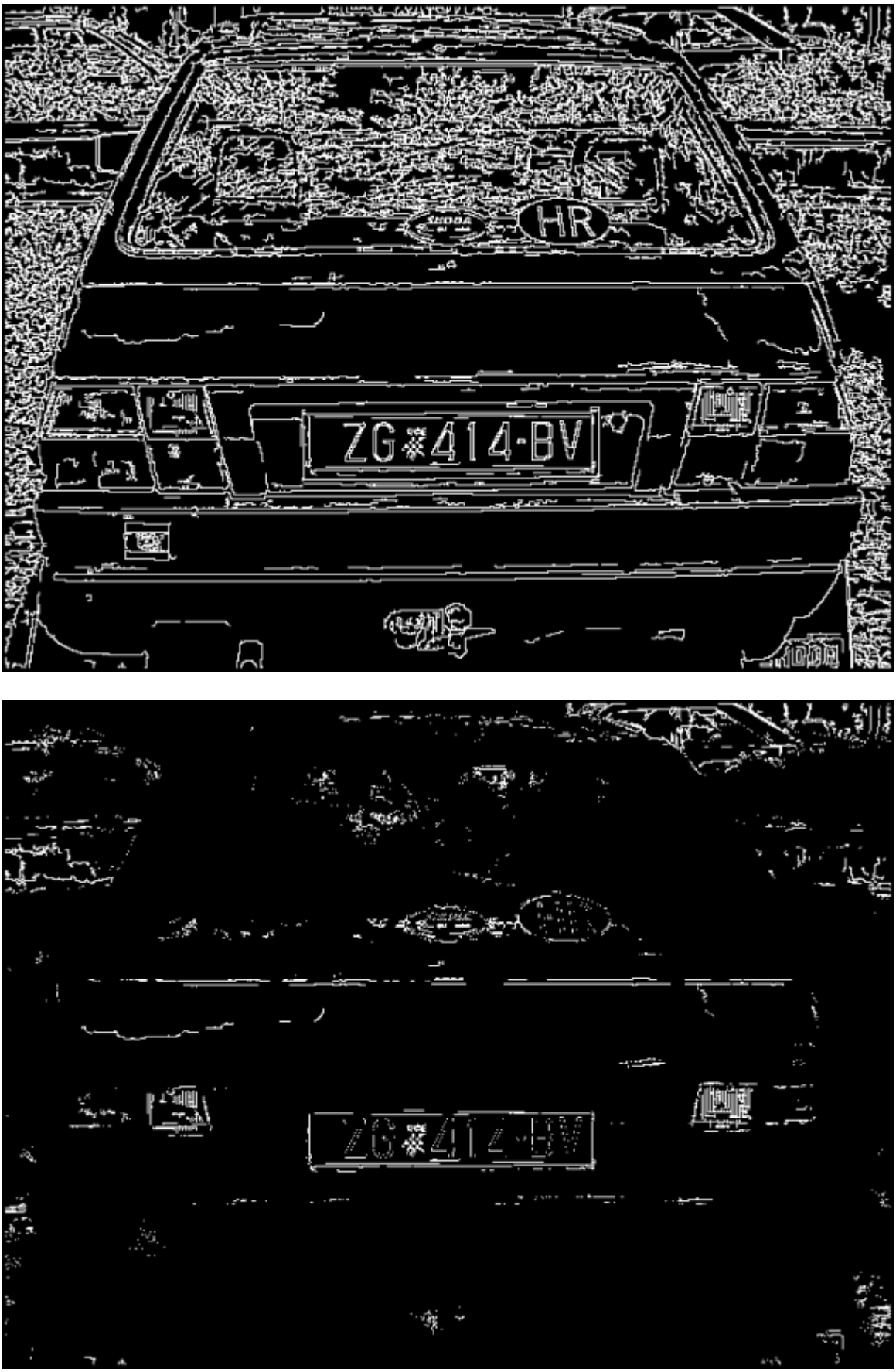
Slika 4.27. Program za preklapanje binarne slike sa detektiranim rubovima.

Kao metode za detekciju rubova koriste se DRF (Difference Recursive Filter) Edge Extract, koja je slična Canny detektoru, Roberts i Sobel.

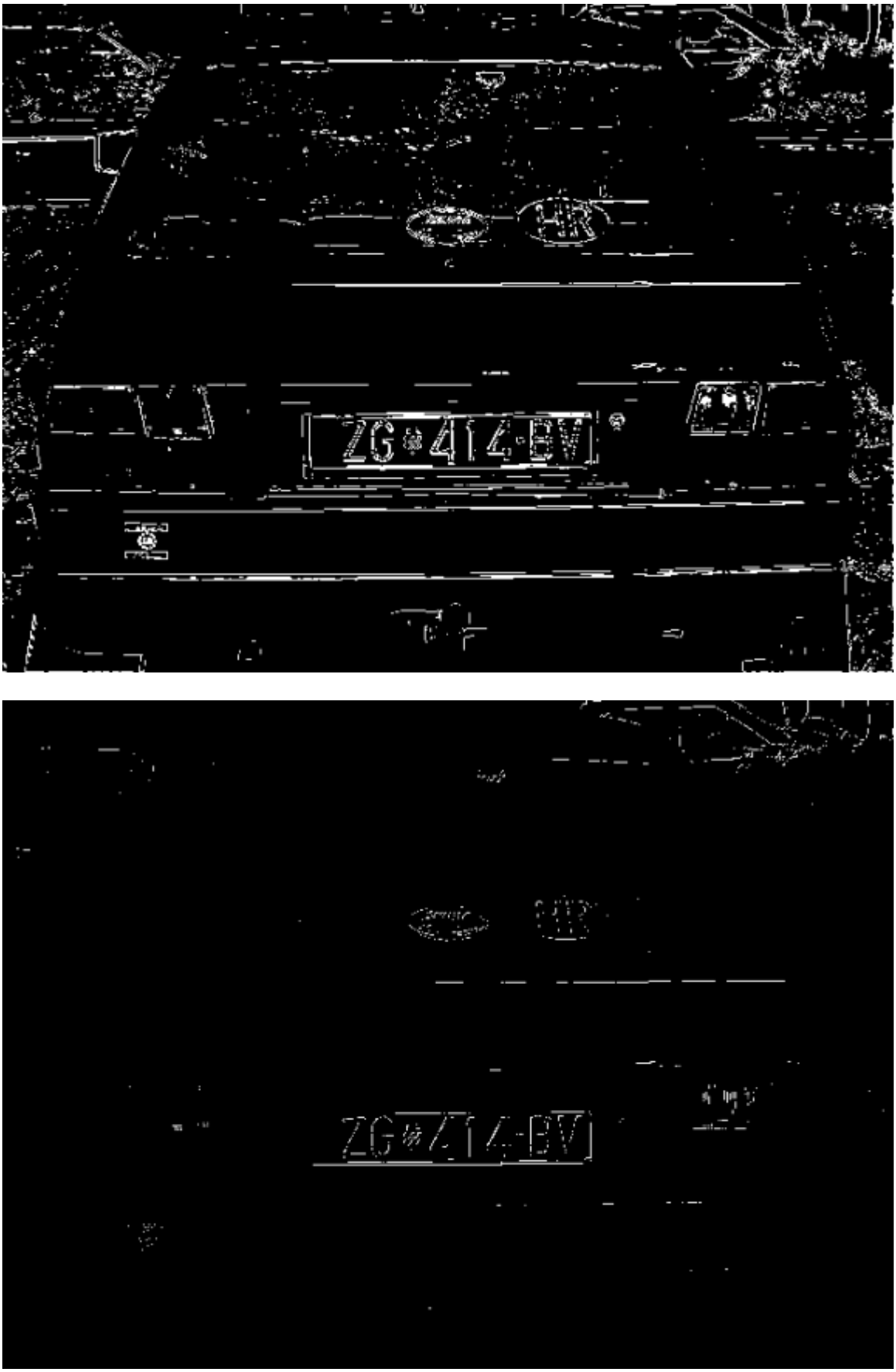
Metode detekcije rubova kao rezultat daju sivu sliku pa se za dobivanje binarne slike rubova koristi Thresh Above.

Kao ulazna slika korištena je RGB slika iz prijašnjih primjera (prikazana je na slici 4.4). Dobivena binarna slika prikazana je na slici 4.7.

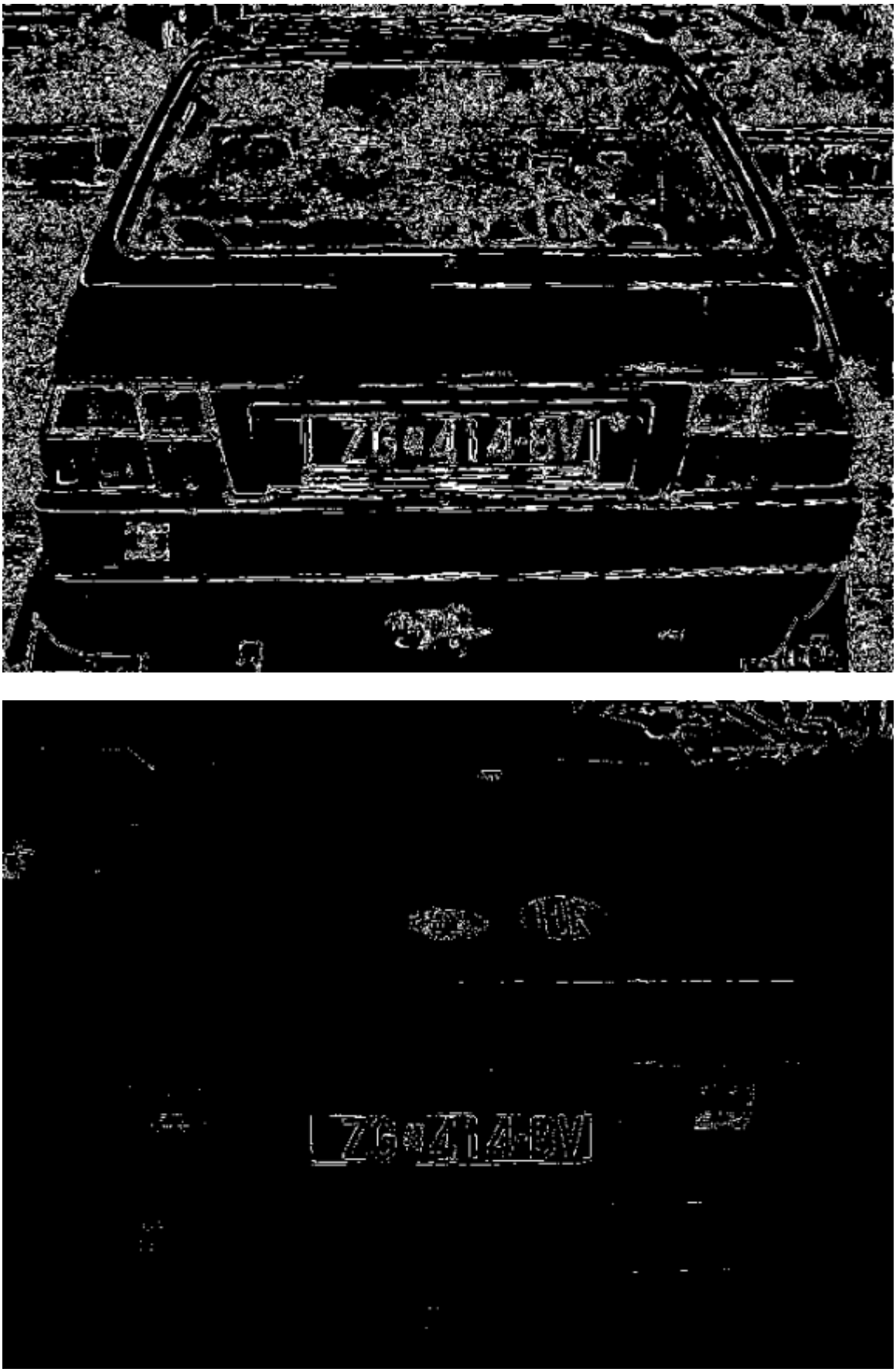
Kod DRF, Roberts i Sobel metoda, Thresh Above prag postavljen je na 200. Dobivene binarne slike rubova i binarne slike rezultata prikazane su slikama.



Slika 4.28. Slika dobivena pomoću metode DRF Edge Extract i preklopljena slika.

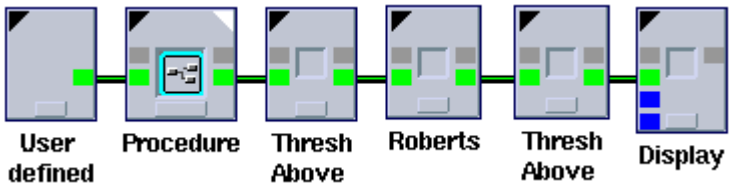


Slika 4.29. Slika dobivena pomoću metode Roberts i preklopljena slika.

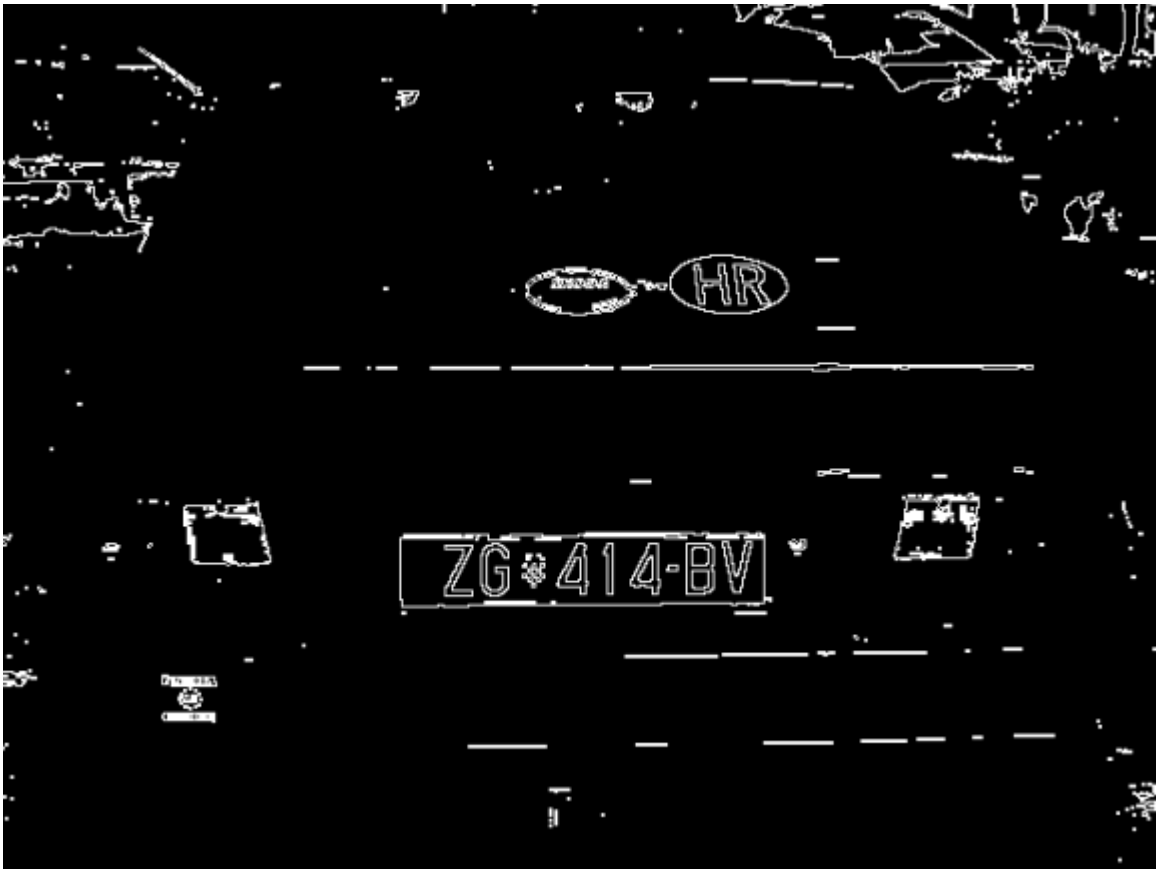


Slika 4.30. Slika dobivena pomoću metode Sobel i preklopljena slika.

Prikazani su i rezultati programa, kod kojeg bi se metoda detekcije rubova direktno primjenila na binarnu sliku. Za prvi Thresh Above prag odabrana je vrijednost 160, a za drugi prag vrijednost 100.



Slika 4.31. Program za primjenu metode Sobel na binarnoj slika.



Slika 4.32. Slika dobivena primjenom metode Sobel na binarnoj slici.

5. Zaključak

Cantata je dobar alat za brzu izradu prototipova. Za izradu programa potrebno je dobro poznavanje postavljenog zadatka i operatora koji mogu poslužiti za riješavanje. Tada je programiranje jednostavno i brzo. Proces programiranja je kratak, nakon odabira Glyphova i nekoliko klikova za povezivanje program je završen.

Provjera programa i praćenje moguće je uz postavljanje Glyphova za prikaz rezultata na više mijesta. Moguće je pokretanje pojedinih Glyphova uz promjenjene parametre i praćenje njihovog utjecaja na krajnji rezultat.

Zbog grafičkog prikaza izvođenja programa, pri čemu Glyphovi izgledom prate promjenu stanja, primjeti se sporost pri izvršavanju, pogotovo ako se program sastoji od petlji ili velikog broja naredbi za upravljanje tokom. Zbog toga je potrebno završeni program prevesti, pri čemu se program zamjenjuje Glyphom. Formira se ekvivalentni operator koji se može pozvati iz komandne linije.

Rezultati problema izdvajanja registarske tabilce vozila iz slike, prikazani su na više načina. Metode poboljšanja sive slike nisu dale bolje rezultate, osim što za različite ulazne slike određuju približno jednaku vrijednost praga za binarizaciju. Najbolji rezultat dobiven je primjenom binarizacije na sivoj slici bez poboljšanja.

Prikazani su i rezultati preklapanja binarne slike sa slikama detektiranih rubova. Najbolji rezultat dobiven je primjenom jedne od metoda za detekciju rubova na binarnoj slici.

Prikazani primjeri služe kao početna faza, nakon čega bi se daljnim postupcima tablica izdvojila iz slike, pa pojedini znakovi tablice, te pokušao interpretirati sadržaj.

Literatura

[1] Khoros Manuals, "Khoros Pro 2001 Overview", PDF dokument, Khoral Inc., 2002.

[2] Khoros Manuals, "Cantata Visual Programming", PDF dokument, Khoral Inc., 2002.

[3] Khoros Manuals, "Toolbox Programming", PDF dokument, Khoral Inc., 2002.

[4] R. Jordan, R. A. Lotufo, "A Hands-on Guide to Khoros Pro 2001", PDF dokument, KRI, ISTEC, 2000.

[5] R. Jordan, R. A. Lotufo, "Digital Image Processing (DIP) with Khoros Pro 2001", HTML dokument, http://www.khoral.com/contrib/contrib/dip2001/index.html, KRI, ISTEC, 2000.

[6] R. Fisher, S. Perkins, A. Walker, E. Wolfart, "Image Processing Learning Resource", HTML dokument, http://www.dai.ed.ac.uk/HIPR2/index.html, 2002.

[7] S. Lončarić, "Digitalna obrada slike: Predavanja", PPT dokument, http://igp.zesoi.fer.hr/teach/dip/index.html, Zagreb, 2001.