საქართველოს ტექნიკური უნივერსიტეტი

მ. თუშიშვილი, ქ. ავალიშვილი

კომპიუტერული გრაფიკა

II ნაწილი

დამტკიცებულია სტუ-ს
სარედაქციო საგამომცემლო
საბჭოს მიერ

თბილისი
2007

განხილულია კომპიუტერული გრაფიკის ერთერთი ძირითადი მიმართულებება - ვექტორული გრაფიკა. მეორე ნაწილში უფრო დაწვრილებით განიხილება მისი ძირითადი პრინციპები და შესაძლებლობები: ობიექტ-ორიენტირებული მიდგომა, ფიგურათა მათემატიკური წარმოდგენა – მათემატიკური მოდელირება, ფრაქტალური გრაფიკის ცნება, გრაფიკული ენები, კერძოდ PostScript ენა, პრიმიტივები და სხვა. განსაკუთრებული ყურადღება ეთმობა კომპიუტერული გრაფიკის მათემატიკურ საფუძვლებს, იგულისხმება ანალიზური გეომეტრიის თეორიის ელემენტები. ანალიზური გეომეტრია განიხილება არა უბრალოდ როგორც წრფივი ალგებრის ნაწილი, არამედ როგორც იმ პრაქტიკულ გეომეტრიულ ამოცანათა ამოხსნის მძლავრი მეთოდოლოგია, რომელიც კომპიუტერულ გრაფიკაში გვხვდება.

განიხილება ასევე ვექტორული და რასტრული გრაფიკის თანამედროვე გრაფიკული რედაქტორები – პროგრამული ინტერფეისი, როგორიც არის პროგრამული პაკეტები: *Corel Draw Graphics Suite X3* და Adobe Creative Suite 2.

აღნიშნული სახელმძღვანელო გათვალისწინებულია საქართველოს ტექნიკური უნივერსიტეტის 22.01. სპეციალობის სტუდენტებისათვის. გარდა ამისა, იგი შეიძლება გამოიყენონ ანალოგიური სპეციალობის სხვა უმაღლესი სასწავლებლების სტუდენტებმაც.

რეცენზენტი: პროფ. ა. ფრანგიშვილი

ISBN 99940-57-69-3 (ყველა ნაწილი)
ISBN 978-99940-949-1-2 (მეორე ნაწილი)

სარჩევი

7. ვექტორული გრაფიკა.

7.1. ვექტორული გრაფიკის ზოგადი ცნებები.

თანამედროვე ვექტორული გრაფიკა – დღეს უკვე აღარ არის ტექსტით განზავებული მარტივი გეომეტრიული ფიგურები. ვექტორული პროგრამა მძლავრ ინსტრუმენტს წარმოადგენს, ის საშუალებას იძლევა შევქმნათ ფოტორეალისტური კოლაჟები. ვექტორულ და რასტრულ გრაფიკას შორის საზღვარი თანდათანობით ქრება და რაც გუშინ მხოლოდ რასტრულ გრაფიკაში იყო შესაძლებელი დღეს ვექტორული პროგრამისათვისაც მისაწვდომია, ამიტომ ვექტორულ პროგრამას, სხვაგვარად, ***ილუსტრატიულ*** რედაქტორს უწოდებენ.

ილუსტრატიული გრაფიკა – კომპიუტერული გრაფიკის გამოყენებითი შტოა, რომელიც ცალკე მიმართულებად ჩამოყალიბდა. მას მიეკუთვნება სურათები, კოლაჟები, რეკლამები, პოსტერები და სხვა, რასაც მხატვრულ პროდუქციას უწოდებენ.

არსებობს ვექტორული გრაფიკის უამრავი პროგრამული პაკეტი, რომელიც შეიცავს, მოხმარების თვალსაზრისით მარტივ, მაგრამ განვითარებულ და მძლავრ ინსტრუმენტულ საშუალებას. მათ რიცხვს მიეკუთვნება მრავლფერიანი გრადიენტული შევსება, გრაფიკული პრიმიტივები – მრავალკუთხედი, ვარსკვლავი სპირალი და სხვა ფორმები, ასევე სპეციალური ეფექტები. თანამედროვე პროგრამები აგრეთვე მრავალფენიანობით ხასიათდება, გარდა ამისა შესაძლებელია ვექტორული გამოსახულების რასტრულში გარდაქმნა და შემდეგ მისი დამუშავება, რაც გულისხმობს ფილტრებისა და მასკების

გამოყენებას. ცხადია ეს ყველაფერი ვექტორული ფაილის ფარგლებში ხდება, და ეს გაცილებით აადვილებს საბოლოო შედეგის მიღებას.

ვექტორული წარმოდგენა გულისხმობს გამოსახულების ელემენტთა მათემატიკური მრუდებით აღწერას, მათი ფერისა და შევსებადობის მითითებით (გავიხსენოთ, რომ წრე და წრეწირი – სხვადასხვა ფიგურაა). მაგალითად, წითელი ელიფსი თეთრ ფონზე აღიწერება სულ ორი, მართკუთხედისა და ელიფსის, მათემატიკური ფორმულით, შესაბამისი ფერების, ზომის და ადგილმდებარეობის მითითებით. *ვექტორულ გრაფიკაში მათემატიკური აპარატი გამოსახულების ვექტორით (წირი და მრუდები) და ასევე ფერისა და მდებარეობის ამსახველი პარამეტრებით აღიწერას გულისხმობს.* მაგალითად, ფოთლის გამოსახულება (სურ.1) აღიწერება იმ წერტილებით, რომელზეც წირი გაივლის, ეს წირი კი გვაძლევს ფოთლის კონტურს. ფოთლის ფერი მოცემულია კონტურისა და ამ კონტურის შიგნით არსებული არის

sur.1.
veqtoruli grafikis magaliTi

ფერით. ცხადია ასეთი აღწერა, რასტრთან შედარებით, გაცილებით მცირე ადგილს იკავებს.

ვექტორული წარმოდგენის კიდევ ერთი უპირატესობაა – ***ხარისხიანი მასშტაბირება***, ობიექტის გადიდება ან შემცირება მათემატიკურ ფორმულებში იწვევს მხოლოდ შესაბამისი კოეფიციენტების გაზრდას ან შემცირებას. ვექტორული გრაფიკის ელემენტთა რედაქტირებისას იცვლება იმ წირისა და მრუდის პარამეტრები, რომლებიც ამ ელემენტთა ფორმას აღწერს. ამ ელემენტთა გადაადგილება, მათი ზომის, ფორმის და ფერის შეცვლა, არ აისახება ვიზუალური წარმოდგენის ხარისხზე. ვექტორული გრაფიკა არ არის დამოკიდებული გადაწყვეტაზე, ე.ი. შეიძლება გადავცეთ სხვადასხვაგვარ გამოტანის მოწყობილობას სხვადასხვა გადაწყვეტით, და ამით ხარისხი არ გაუარესდება. გარდა ამისა არსებობს მოწყობილობათა გარკვეული კლასი, რომელიც ორიენტირებულია კონკრეტულად ვექტორულ მონაცემთა ასახვაზე. მათ მიეკუთვნებათ გრაფოამგები და ზოგიერთი ლაზერული პროექტორი.

ვექტორულ პროგრამაში გაცილებით მოსახერხებელია ტექსტთან მუშაობა, გეომეტრიული ფიგურების შექმნა, უფრო მარტივადაა ორგანიზებული ფერთან მუშაობაც. კომპიუტერის სიმძლავრის ზრდასთან ერთად ბევრ რასტრულ რედაქტორს ვექტორულ გრაფიკასთან მუშაობის დამატებითი ფუნქცია გაუჩნდა. იგივე მიზეზით, ვექტორულ გრაფიკაშიც გახდა შესაძლებელი ფოტორეალისტური ეფექტების შექმნა, როგორიცაა ჩრდილი, გამჭვირვალეობა, მოცულობითი ეფექტები და სხვა.

ორ- და სამგანზომილებიანი გრაფიკის მარტივი ობიექტებია: რკალი, წირი, წრეხაზი, სფერო, კუბი, და ა.შ., მათ ***პრიმიტივები*** ეწოდებათ და ისინი უფრო რთული ობიექტის შესაქმნელად გამოიყენება. გამოსახულება, ვექტორულ გრაფიკაში იქმნება სხვადასხვა ობიექტების კომბინაციით, მაგალითად ოთხკუთხედი შეიძლება განვიხილოთ როგორც ოთხი წირის კომბინაცია, ხოლო ობიექტი კუბი, რომელიც უფრო რთულია: შეიძლება განვიხილოთ როგორც თორმეტი დაკავშირებული წირი, ან ექვსი დაკავშირებული ოთხკუთხედი (სურ.2).

სურ.2. ვექტორული პრიმიტივების გამოყენების მაგალითები

ყველა ობიექტს ატრიბუტები (თვისებები) გააჩნია. ამ თვისებებს მიეკუთვნება: წირის ფორმა, მისი სისქე, ფერი, წირის ხასიათი (უწყვეტი, წყვეტილი და ა.შ.). ჩაკეტილი წირი, დამატებით ხასიათდება შევსებით. არე, ჩაკეტილი კონტურის შიგნით, შეიძლება შევავსოთ ფერით ან ტექსტურით. ვექტორული ობიექტის აღწერის რეალურ ბრძანებებს ცხადია ვერ ვხედავთ. ამის განსაზღვრა დაკისრებული აქვს იმ კომპიუტერულ პროგრამას (მაგ. Corel Draw), რომელსაც გამოსახულების შესაქმნელად

ვირჩევთ. მაგალითად, წრეხაზის აღწერა PostScript ენაზე შესაძლებელია ერთი სტრიქონით:

Object 2076.19 2548.51 2220.84 2693.16 @E

როგორც უკვე აღვნიშნეთ, ამ ბრძანებას მომხმარებელი ვერ დაინახავს, მას ჰქმნის გამოყენებული პროგრამა. ამ მაგალითიდან ჩანს, რომ წრეხაზის ვექტორული გამოსახულება, რასტრთან შედარებით, გაცილებით ნაკლებ ადგილს იკავებს მეხსიერებაში.

ვექტორული გრაფიკა, ობიექტის აღწერისათვის, კომპიუტერულ ბრძანებებს და მათემატიკური ფორმულების კომბინაციას იყენებს, რაც მისი საკვანძო მომენტია.

ვექტორული ობიექტი მეხსიერებაში პარამეტრების ნაკრების სახითაა წარმოდგენილი, მიუხედავათ ამისა არ უნდა დაგვავიწყდეს, რომ ეკრანზე ყველა გამოსახულება მაინც წერტილებით აისახება, იმ უბრალო მიზეზის გამო, რომ ეკრანი ასეა მოწყობილი. ყოველი ობიექტის ეკრანზე გამოტანა უკავშირდება მისი ეკრანული გამოსახულების წერტილთა კორდინატების გამოთვლას, ამიტომ ვექტორულ გრაფიკას ზოგჯერ ***გამომთვლელ გრაფიკასაც*** უწოდებენ. ანალოგიური გამოთვლები სრულდება ობიექტის პრინტერზე გატანის დროსაც.

ვექტორული გრაფიკის ფაილი შეიძლება რასტრულ გამოსახულებასაც შეიცავდეს, როგორც ერთ-ერთი ტიპის ობიექტს, რადგან რასტრული სურათი კომპიუტერისთვის უბრალოდ ინსტრუქციათა ნაკრებია. თუმცა აქვე უნდა აღვნიშნოთ, რომ რასტრის რედაქტირება და ცალკეულ პიქსელთა შეცვლა ვერ ხერხდება, შესაძლებელია

მხოლოდ ზომისა და მასშტაბის შეცვლა ან კონტრასტისა და სიკაშკაშის რეგულირება.

ვექტორული გრაფიკის ფაილი სხვადასხვა ელემენტს შეიცავს: ვექტორულ ბრძანებათა ნაკრებს – გამოსახულების შესაქმნელად; ინფორმაციულ ცხრილს – ნახატის ფერის შესახებ; მონაცემებს, იმ შრიფტების შესახებ, რომელიც ნახატზეა გამოყენებული და სხვა.

ზოგიერთი ვექტორული ფორმატი მარტივია და რამოდენიმე ათეულ ბრძანებას შეიცავს, სხვა ფორმატში კი ბრძანებათა რიცხვი შესაძლოა ათობით და ათასობით იზომებოდეს.

სხვადასხვა ვექტორულ ფორმატს, სხვადასხვა ფერთა შესაძლებლობა აქვს. უმარტივესი ფორმატი, შეიძლება საერთოდ არ შეიცავდეს ინფორმაციას ფერის შესახებ, და იყენებდეს იმ მოწყობილობათა ფერს, რომელზეც ხდება მათი გატანა. არსებობს უფრო რთული ფორმატი, რომელსაც შეუძლია ფერის შესახებ 32-ბიტიანი მონაცემის შენახვა. თუმცა, ეს ფაილის ზომაზე გავლენას არ ახდენს (რასაკვირველია, თუ ფაილის შიგნით არ არის რასტრული ობიექტი). ჩვეულებრივ ვექტორულ ობიექტში, როგორიცაა წრეწირი, კვადრატი, და ა.შ., ფერი ეკუთვნის მთლიანად მთელ ობიექტს. ობიექტის ფერი ინახება, როგორც მისი ვექტორული აღწერილობის ნაწილი.

ზოგიერთ ვექტორულ ფაილს გამოსახულების რასტრული ესკიზის შექმნაც შეუძლია, რომელიც ფაილშივე ინახება. ეს შეიძლება სასარგებლო იყოს ისეთ სიტუაციაში, როცა არ გვინდა მთელი ფაილის გახსნა და მხოლოდ იმის

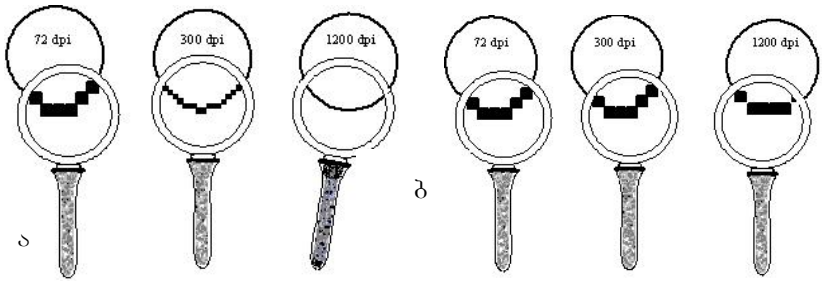
დანახვა გვინდა რა ინახება მასში, როცა შეუძლებელია ვექტორული სურათის დანახვა (მაგალითად, ერთერთ საგამომცემლო პაკეტში, არ იგზავნება სურათი ბეჭდვაზე). ცხადია, ეს უპირატესობა ზრდის მეხსიერებას, რადგან რასტრი დიდი მოცულობით ხასიათდება.

სამწუხაროდ, ვექტორული ფორმატის გადაცემა არამომგებიანი ხდება მაშინ, როცა გამოსახულება ძალიან გადატვირთულია ფერთა ბევრი ტონით ან ბევრი მცირე ზომის დეტალით. ეს დეტალები, ***გრაფიკული პრიმიტივების*** ერთობლიობას წარმოადგენს, და ყოველი მათგანი ფორმულას შეესაბამება. გრაფიკული პრიმიტივების ქვეშ იგულისხმება გრაფიკული ობიექტბების ის მინიმალური რაოდენობა, რომელიც ვექტორულ სურათს შეადგენს – იმ აგურების მსგავსად, რითიც შენდება შენობა. ამას კი ისევ მივყავართ ფაილის ზომის გაზრდასთან.

ვექტორული გრაფიკის უპირატესობა:

- იყენებს გამოტანის მოწყობილობათა გადაწყვეტას და ყველა მის უპირატესობას, რაც ვექტორული სურათის, ხარისხის დანაკარგის გარეშე, ზომის ცვლილების საშუალებას იძლევა. ვექტორული ბრძანება გამოტანის მოწყობილობას უბრალოდ ამცნობს, რომ საჭიროა მოცემული ზომის ობიექტის დახატვა და ამისათვის იმდენი წერტილი გამოიყენოს რამდენიც შესაძლებელია. რასტრული ფორმატი კი, ვექტორულისგან განსხვავებით, ზუსტად განსაზღვრავს, რამდენი პიქსელი უნდა შეიქმნას და ეს რაოდენობა არ იცვლება, მოწყობილობის გადაწყვეტის ცვლილებასთან ერთად. პრინტერის

გადაწყვეტის გაზრდის დროს ამას მივყავართ ან გამოსახულების ზომის შემცირებასთან, ან ყოველი პიქსელისთვის წერტილების უფრო მეტი რაოდენობის გამოიყენებასთან. მე-3 სურათზე ნაჩვენებია წრეწირის რასტრული და ვექტორული გამოსახულების შედარება.



სურ.3. წრეწირის ბეჭდვა, სხვადასხვა გადაწყვეტის მქონე პრინტერზე:
ა – ვექტორული, ბ – რასტრული.

- სურათის ცალკეულ ნაწილთა რედაქტირება, გავლენას არ ახდენს დანარჩენ ნაწილებზე (რასტრული გამოსახულების შემთხვევაში კი მოგვიწევდა ყოველი პიქსელის რედაქტირება).
- ვექტორული გამოსახულება, რომელიც რასტრულ ობიექტს არ შეიცავს, კომპიუტერში შედარებით მცირე ადგილს იკავებს (10/1000-ჯერ უფრო მცირეს, ვიდრე მისი რასტრული ანალოგი).

7.2. ობიექტ-ორიენტირებული მიდგომა

ვექტორული გრაფიკის პროგრამები ობიექტ-ორიენტირებულ მიდგომაზეა დაფუძნებული. რაც შეეხება ტერმინს *"ობიექტ-ორიენტირებული"* ის ასე უნდა გვესმოდეს – ყველა ოპერაცია, რომელიც მომხმარებელის მიერ გამოსახულების შექმნისა და დამუშავების პროცესში სრულდება, ტარდება არა მთლიანად გამოსახულებაზე ან მის უმცირეს ნაწილაკებზე (წერტილოვანი გამოსახულების პიქსელებზე) არამედ ობიექტზე – გამოსახულების შემადგენელ სემანტურ ელემენტებზე. მომხმარებელს შეუძლია ააგოს როგორც სტანდარტული (წრე, მრავალკუთხედი, ტექსტი და სხვა.) ისე შედგენილი ობიექტი და შემდგომში განიხილოს და იმოქმედოს მასზე, როგორც ერთ მთლიანზე. ამგვარად გამოსახულება იერარქიული სტრუქტურა ხდება, რომლის სათავე ილუსტრაციაა, ხოლო ყველაზე ქვედა ელემენტი – სტანდარტული ობიექტი.

ასეთი მიდგომისას, ობიექტთა ყოველ სტანდარტულ კლასს შეესაბამება მმართველი პარამეტრების, ან ატრიბუტების უნიკალური ერთობლიობა. ***მაგალითად,*** თუ ვლაპარაკობთ მართკუთხედზე, რომლის სიმაღლეა 200მმ., სიგანე 300მმ., შევსებულია ლურჯი ფერით, კონტური ყვითელია, სისქე 3 პუნქტი და ცენტრის მდებარეობის კოორდინატებიც განსაზღვრულია, მაშინ საქმე გვაქვს ობიექტთან, ანუ გარკვეული კლასის ეგზემპლიართან, რომლის მმართველი პარამეტრები დაფიქსირებულია. ნებისმიერი ზომისა და ფორმის კონტური ძალიან ზუსტად აისახება მათემატიკური მოდელის საშუალებით, რადგან ის წერტილებითა და მრუდებით ფორმირდება, და

არა მართკუთხა პიქსელებით შევსებული ბადით, როგორც ეს რასტრულ გამოსახულებაშია. აქედან გამომდინარეობს ობიექტ-ორიენტირებული მიდგომის კიდევ ერთი თავისებურება – ობიექტთა ყოველი სტანდარტული კლასისთვის განსაზღვრულია სტანდარტულ ოპერაციათა ჩამონათვალი. მაგალითად, ზემოთ აღწერილი მართკუთხედი შეიძლება მოვაბრუნოთ, შეუცვალოთ მასშტაბი, მოუმრგვალოთ კუთხეები, გარდავქმნათ სხვა კლასის ობიექტად – ჩაკეტილ მრუდად. ობიექტზე ორიენტაცია მუშაობის პროცესს თითქმის შეუზღუდავ მოქნილობას ანიჭებს.

ობიექტ-ორიენტირების კონცეფციით აგებული პროგრამული პაკეტის ლოგიკური დასრულებაა ობიექტ-ორიენტირებული პროგრამირების ენის, VBA-ს შემოტანა. ამ ენაზეა აგებული პროგრამული მოდულები, რაც ხშირად გამეორებადი მოქმედებების ვტომატიზების საშუალებას იძლევა. შესაძლებელია ასევე სპეციალიზირებული გრაფიკული სისტემების აგება, ობიექტთა ახალი კლასისა და მათზე შესრულებადი ოპერაციების განსაზღვრით. უფრო მეტიც, ეს პროცესი ავტომატიზებულია. VBA-ზე შექმნილ პროგრამულ მოდულში მოქმედებათა თანმიმდევრობის გარდაქმნა და მათი ავტომატურად ჩაწერაც შესაძლებელია.

გამოსახულების ერთხელ უკვე აგებული ფრაგმენტი, შეიძლება სხვა ნამუშევარშიც გამოვიყენოთ, რადგან ეს ფრაგმენტი სპეციალურ ბიბლიოთეკაში ინახება. შესაძლებელია ასევე გამოსახულების შეუცვლელად, სამუშაოს ხასიათის შევცვლაც. მაგალითად, რეკლამური პოსტერი რომ გადავაქციოთ ყავის ჭიქაზე, ან საწერ

კალამზე ნაწებად, ან ანიმაციურ გამოსახულებად web-ისთვის, საჭიროა მცირე დამატებითი სამუშაოს შესრულება.

ვექტორულ პროგრამათა (მაგალითად, CorelDRAW) აღნიშნული თავისებურება ამარტივებს და სტრუქტურირებულს ხდის მათ შესწავლასაც. კერძოდ, თუ გვეცოდინება რამოდენიმე ობიექტის ატრიბუტი და მასზე შესრულებადი ოპერაცია, მუშაობის დაწყება პრაქტიკულად შესაძლებელია.

7.3. ფრაქტალური გრაფიკის ცნება

ფრაქტალური გრაფიკის პროგრამული საშუალებები გამოსახულებათა ავტომატური გენერაციისთვისაა განკუთვნილი და მათემატიკური გამოთვლების გზით მიიღება. ამიტომ, ფრაქტალური მხატვრული კომპოზიციის შექმნა დაკავშირებულია პროგრამირებასთან და არა ხატვასთან ან გაფორმებასთან. ფრაქტალურ გრაფიკას იშვიათად იყენებენ ბეჭდური ან ელექტრონული დოკუმენტების შესაქმნელად, მაგრამ ხშირად იყენებენ გასართობ პროგრამებში.

ფრაქტალური გრაფიკა, ვექტორულის მსგავსად – გამოთვლითია, მაგრამ მისგან განსხვავებით, არანაირ ობიექტს არ ინახავს კომპიუტერის მეხსიერებაში. გამოსახულება აიგება განტოლების (ან განტოლებათა სისტემით) მიხედვით, ამიტომ ფორმულების გარდა არაფრის შენახვა არ არის საჭირო. განტოლებაში კოეფიციენტთა შეცვლით, სრულიად განსხვავებული სურათების მიღება შეიძლება.

1. უმარტივესი ფრაქტალური ობიექტია – ფრაქტალური სამკუთხედი. ააგეთ ჩვეულებრივი ტოლგვერდა სამკუთხედი a გვერდით. დაყავით ყოველი გვერდი სამ მონაკვეთად. გვერდის შუა მონაკვეთზე ააგეთ ტოლგვერდა სამკუთხედი, რომლის გვერდი საწყისი სამკუთხედის გვერდის მესამედია – a/3. დანარჩენ მონაკვეთებზე ააგეთ ტოლგვერდა სამკუთხედები, რომელთა გვერდები a/9 –ის ტოლია. მიღებულ სამკუთხედებზე გაიმეორეთ ანალოგიური ოპერაციები. მალე თქვენ დაინახავთ, რომ შემდეგი თაობის სამკუთხედები, მემკვიდრეობით იღებენ უფროსი თაობის სტრუქტურის თვისებებს. ასე იბადება ფრაქტალური ფიგურა.

სურ.4. ფრაქტალურ ფიგურათა მაგალითები.

2. მემკვიდრეობის პროცესი შეიძლება უსასრულოდ გაგრძელდეს. ასეთი უსასრულო ფრაქტალური ობიექტის ლუპით ან მიკროსკოპით განხილვისას, მასში ვიპოვით სულ ახალ-ახალ დეტალებს, რომლებიც საწყისი სტრუქტურის თვისებებს იმეორებს.

3. ფრაქტალურ თვისებათა მატარებელია, ცოცხალი და არა ცოცხალი ბუნების ბევრი ობიექტი. მაგალითად, ჩვეულებრივი ფიფქი, მისი მრავალჯერადი გადიდება დაგვარწმუნებს მის ფრაქტალურ ბუნებაში. ფრაქტალური ალგორითმები ჩადებულია კრისტალებისა და მცენარეების ზრდაშიც. შეხედეთ მცენარე გვიმრის ტოტებს, და დაინახავთ, რომ ყოველი შვილობილი ტოტი იმეორებს უფრო მაღალი დონის ტოტის თვისებებს.

4. ხის ცალკეულ ტოტში, წმინდა მათემატიკური მეთოდებით, შეიძლება თვალყური ვადევნოთ მთელი ხის თვისებებს. ხოლო თუ ტოტს წყალში ჩავდებთ, ის მალე ფესვებს გამოიტანს და გარკვეული დროის შემდეგ სრულფასოვან ხედ გადაიქცევა.

5. ფრაქტალური გრაფიკის ამ ნიჭს, ხშირად იყენებენ ცოცხალი ბუნების მათემატიკური მოდელირებისას, კერძოდ, განსაკუთრებული ილუსტრაციების ავტომატური გენერაციისთვის.

7.4. გრაფიკული ენები.

კომპიუტერულ გრაფიკაში ნებისმიერი გრაფიკული ოპერაცია, ფაქტიურად ცალკეულ პიქსელთან მუშაობას გულისხმობს, თუნდაც მოცემული ფერის წერტილი დავსვათ ან მოცემული წერტილის ფერი განვსაზღვროთ. ბიბლიოთეკების უმრავლესობა კი, უფრო რთულ ობიექტებთან ურთიერთობს, რადგან ცალკეულ პიქსელებთან მუშაობა პროგრამისტისათვის ძალიან რთული და არაეფექტურია.

ასეთ ობიექტებს შორის, რომელიც პიქსელთა ერთობლიობას წარმოადგენს, შეიძლება გამოვყოთ შემდეგი ძირითადი ჯგუფები:

- წრფივი გამოსახულება (წირთა რასტრული სახე);
- მთლიანი (უწყვეტი) ობიექტი (ორგანზომილებიანი არის რასტრული სახე);
- შრიფტი;
- გამოსახულება (პიქსელთა მართკუთხა მატრიცა).

როგორც წესი, ყოველ კომპილატორს აქვს საკუთარი გრაფიკული ბიბლიოთეკა, რომელიც გრაფიკული ობიექტების ძირითად ჯგუფებთან მუშაობს.

მაგალითად, გრაფიკული ბრძანებები აქვთ Basic, Borland C++, Pascal ენებს. Lisp პროგრამირების ენა AutoCAD-ის ბაზაზე აგებულ სისტემათა პროგრამირებაშია გამოყენებული. საკმაოდ საინტერესოა PostScript ენაც, რომელიც აღწერს გვერდებს.

PostScript იქმნებოდა, როგორც მარტივი სტანდარტული ენა. მისი დანიშნულებაა ტექსტის, სურათის და მარტივი გამოსახულების ბეჭდვით ფურცელზე აღწერა. ამან თავისებური დაღი დაასვა PostScript-ში გამოყენებულ კონსტრუქციებს. ენა შეიცავს 250-მდე ოპერატორს, რაც ერთი და იგივე მოქმედების სხვადასხვა გზით დავაპროგრამების საშუალებას იძლევა. მეორეს მხრივ ითვლებოდა, რომ PostScript-ის პროგრამები დიდი არ უნდა ყოფილიყო, ამიტომ ამ ენაში სტრუქტურირების საშუალებები არც თუ ისე განვითარებულია. PostScript ენის მეასმედი გრაფიკას ეძღვნება, დანარჩენი კი პროგრამირების ჩვეულებრივი პროცედურული ენაა, რომელიც ბევრი სხვა ენის ელემენტს შეიცავს. ძალიან მნიშვნელოვანია, რომ ფურცლების აღწერა PostScript-ში

არ არის დამოკიდებული პერიფერიულ მოწყობილობებზე. როგორც წესი პროგრამები PostScript-ზე დანართებით გენერირდება, მაგალითად ტექსტური პროცესორებით, საგამომცემლო სისტემის პროგრამებით, კერძოთ CorelDraw.

PostScript ენის შექმნისა და განვითარების ერთ-ერთი მიზეზი ტექსტისა და გამოსახულების რასტრულ გარე მოწყობილობებზე (მატრიცული, ჭავლური და ლაზერული პრინტერები და ასევე მონიტორები) გამოტანის აუცილებლობაა. როგორც უკვე ვიცით, ამ მოწყობილობებზე გამოსახულება პიქსელთა მასივებისაგან იწყობა – ეს მათი ძირითადი თვისებაა. ყოველ პიქსელს ამ მასივში თავისი მისამართი, სტრიქონისა და კოლონის ნომერი, და ასევე ფერი აქვს.

PostScript მოწყობილობას (პრინტერი, მონიტორი), PostScript ენის ინტერპრეტატორი გააჩნია, რომელიც კომპიუტერიდან ფურცლის აღწერილობის ტექსტურ ფაილს ღებულობს, რომელსაც შემდეგ რასტრულ ფორმაში გარდაქმნის და პრინტერზე ან ეკრანზე გამოიტანს. აღწერილობის გარდაქმნისას ინტერპრეტატორი სხვადასხვა მოქმედებას ასრულებს, რაც მოწყობილობის ტიპზეა დამოკიდებული. მაგალითად, თუ შავთეთრ საბეჭდ მოწყობილობაზე ნახევარტონიანი გამოსახულება უნდა გამოვიტანოთ, მაშინ PostScript ინტერპრეტატორი ამ მოწყობილობის ნახევარტონიან არეებს ზონებად დაყოფს. ეს ზონები კი ნაცრისფერის განსაზღვრულ დონეებს შეესაბამება, ამის მიხედვით ხდება ამ ზონებში შავი წერტილების სიმკვრივის

განსაზღვრა. სვადასხვა სიმკვრივის შავი ფერის წერტილები კი ნახევარტონების ილუზიას ქმნის.

სხვადასხვა მიზეზიდან გამომდინარე, გაცილებით მომგებიანია ფურცლის აღწერის, და არა მისი რასტრული წარმოდგენის ანუ პიქსელების მთელი მასივის გადაცემა. პირველი – ფურცლის აღწერა, ბევრად უფრო კომპაქტურია, ვიდრე კარგად დაარქივებული გამოსახულება. მეორე – ბეჭდური ფურცლის მომზადება საკმაო დროს ითხოვს, აღწერის გადაცემის შემთხვევაში, ბეჭდვისთვის მზადება პრინტერს ან დისპლეის პროცესორს დაეკისრება, ამ დროს კი კომპიუტერის პროცესორი თავისუფლდება და შეუძლია მუშაობის გაგრძელება, ცხადია ამით მისი მწარმოებლობაც გაიზრდება. და ბოლოს ყველაზე მნიშვნელოვანი უპირატესობა – აღწერა არ არის დამოკიდებული გარე მოწყობილობის ტიპზე.

PostScript ენაში შეტანილია მიმდინარე ფურცლის ცნება, რაც ფურცლის აღწერის დამოუკიდებლობას უზრუნველყოფს. *მიმდინარე ფურცელი* "იდეალური" ფურცელია მეხსიერებაში, რომელზეც PostScript-ი ხატავს. ის არ არის დამოკიდებული იმ კონკრეტული პრინტერის ფიზიკურ მახასიათებლებზე, რომელზეც ის დაიბეჭდება. პროგრამის მუშაობის დაწყებისას ეს ფურცელი სუფთაა, როცა მიმდინარე ფურცელი მთლიანად აღიწერება, ის იგზავნება პრინტერზე და იბეჭდება ისეთი ხარისხით, რა შესაძლებლობებიც კონკრეტულ პრინტერს გააჩნია.

მიმდინარე გზა (current path) ეს არის ერთმანეთთან დაკავშირებული ცალკეული წერტილების, წირების,

მრუდების ნაკრები, რომელიც ფიგურებს და მათ მდებარეობას აღწერს. მიმდინარე გზა არანაირ შეზღუდვებს არ ექვემდებარება. მაგალითად, შეუძლია თავის თავის გადაკვეთა და ა.შ. მიმდინარე ფურცელზე მიმდინარე გზის ელემენტები პოზიციითაა მოცემული.

ჩამონაჭრების მიმდინარე გზა (Current clipping path) ეს იმ არის საზღვრებია, რომელშიც შეიძლება გამოსახულება იყოს დახატული.

კოორდინატთა სისტემა. ელემენტის პოზიცია ფურცელზე აღიწერება წყვილი კოორდინატით x,y. გამოტანის ყოველ მოწყობილობას თავისი კოორდინატთა სისტემა აქვს, რომლის საშუალებითაც ხდება ფურცილზე წერტილის დამისამართება. ამ სისტემას მოწყობილობის სივრცე (device space) ეწოდება, რომელიც ყველა მოწყობილობისთვის განსხვავებულია. არ არსებობს ერთგვარობა კოორდინატთა სათავის მდებარეობისა და ასევე ვერტიკალურ და ჰორიზონტალურ ღერძებზე მასშტაბირებაში.

მიმდინარე PostScript ფურცელზე, პოზიცია მომხმარებლის კოორდინატთა სისტემაში (ან მომხმარებლის სივრცეში) აღიწერება, რომელიც არ არის დამოკიდებული მოწყობილობის სივრცეზე. PostScript – პროგრამაში მიმდინარე ფურცლის ბეჭდვის წინ, კოორდინატები მომხმარებლის სივრციდან ავტომატურად გარდაიქმნება მოწყობილობის სივრცეში. ამგვარად, მომხმარებლის სივრცე ისეთ კოორდინატთა სისტემას წარმოადგენს, რომლის შიგნითაც ფურცელი კონკრეტული ბეჭდვის მოწყობილობისგან დამოუკიდებლადაა აღწერილი.

მომხმარებლის სივრცე შეიძლება სამი ხერხით შეიცვალოს:

1. კოორდინატთა სისტემის სათავე შეიძლება გადავიტანოთ მომხმარებლის სივრცის ნებისმიერ წერტილში;
2. ღერძები შეიძლება შემობრუნდეს ნებისმიერი მიმართულებით;
3. მასშტაბი ნებისმიერად შეიძლება ვცვალოთ, ყოველი ღერძის მიმართ, ე.ი. შესაძლებელია ნებისმიერი წრფივი გარდაქმნის შესრულება მომხმარებლის სივრციდან მოწყობილობის სივრცეში.

სიმბოლოთა ნაკრები. ყველა სახის ფრჩხილი (მრგვალი, კვადრატული, ფიგურული, კუთხოვანი) და პროცენტის ნიშანი, PostScript-სათვის სპეციალური ნიშანია. ASCII კოდის ქვესიმრავლის ყველა დანარჩენი სიმბოლო შეზღუდვის გარეშე გამოიყენება პროგრამაში და რეგულარული სიმბოლო ეწოდება.

სტეკი. მონაცემთა დამუშავებისათვის PostScript-ს მეხსიერების ნაწილი აქვს დარეზერვებული, რომელსაც სტეკი ეწოდება. სტეკში თავსდება მონაცემები და იქიდან შებრუნებული მიმდევრობით ამოდის ე.ი. პირველად იშლება ბოლოს ჩაწერილი რიცხვი. მომსახურების ასეთ ფორმას LIFO ეწოდება. სინამდვილეში PostScript ოპერირებს ოთხი სხვადასხვა სტეკით: ოპერანდების, ლექსიკონის, შესრულების და გრაფიკის მდგომარეობის.

ოპერანდების სტეკი შეიცავს PostScript-ის ობიექტებს და მათზე ჩატარებული მოქმედების შედეგებს.

PostScript-ის ოპერატორი ოპერანდებს მხოლოდ სტეკიდან ღებულობს. მონაცემებზე ოპერაციის ჩატარებისას მოითხოვება, რომ მათი ოპერანდები თავდაპირველად სტეკში მოთავსდეს. პროგრამირების ასეთ სტილს, როცა ოპერანდები ოპერატორზე (მათზე ჩასატარებელ ოპერაციებზე) წინ არის მოცემული, წინასწარ ფიქსირებული ნოტაცია ეწოდება. მაგალითად, ორი რიცხვის (4 და 5) შეკრების ოპერაცია PostScript-ში ასე გამოიყურება – 4 5 add.

მონაცემთა ტიპები. პროგრამირების როგორც სხვა გავრცელებულ ენებში, PostScript-შიც სხვადასხვა ტიპის მონაცემები განიხილება: integer, real, boolean, მასივები და სტრიქონები. განისაზღვრება აგრეთვე mark (ნიშნული, ჭდე) ტიპის ობიექტები და dictionary (ლექსიკონი).

PostScript-ში მონაცემთა ელემენტებს (რიცხვებს, მასივებს, სტრიქონებს, სიმბოლოებს) ობიექტი ეწოდება. ობიექტზე შეიძლება სხვადასხვა მოქმედება შესრულდეს, მაგრამ თუ სხვა ენაში ობიექტი ცვლადებში თავსდება და ცვლადის სახელის მითითებით მისამართდება, სტეკის მექანიზმის გამოყენებით, PostScript პირდაპირ მონაცემებთან მუშაობს. მაგალითად, თუ გვაქვს სტრიქონი -8 10.4 +77..., მაშინ ინტერპრეტატორი მარცხნიდან მარჯვნივ ამოკითხვით შემდეგ მოქმედებებს ასრულებს:

1. სტეკში შეიტანს რიცხვს–8-ს; გადაადგილებს მიმმართველს შემდეგ თავისუფალ პოზიციაზე.

2. სტეკში შეიტანს რიცხვს 10.4-ს; გადაადგილებს მიმმართველს შემდეგ თავისუფალ პოზიციაზე და ა.შ.

ოპერაციის შესრულებისას თავდაპირველად იყენებს ბოლოს შეტანილ რიცხვს 77-ს, რადგან ის სტეკის სათავეშია. დანარჩენ რიცხვებს კი იყენებს სტეკის პრინციპით.

PostScript პროგრამაში პრობელი, ტაბულაციისა და ახალი სტრიქონის სიმბოლო ობიექტთა დამყოფს წარმოადგენს, ზოგიერთი სხვა სიმბოლოც, როგორიცაა მრგვალი და ოთხკუთხედი ფრჩხილი, ზოგჯერ შეიძლება ასევე დამყოფს წარმოადგენდეს.

PostScript-ში *ოპერატორი* სიტყვაა, რომელიც ინტერპრეტატორს აიძულებს შეასრულოს ესა თუ ის მოქმედება. პროგრამირების სხვა ენებში ეს ბრძანებისა და პროცედურის ექვივალენტია. როცა ინტერპრეტატორი PostScript პროგრამაში სიტყვას აღმოაჩენს, ცდილობს გაარკვიოს, არსებობს თუ არა მისი შესატყვისი ოპერატორის სახელი ლექსიკონში, თუ ეს სიტყვა ლექსიკონში მოიძებნა, მაშინ სრულდება ყველა მასთან დაკავშირებული მოქმედება და გადადის ფაილში არსებულ შემდეგ სიტყვაზე.

add და sub ოპერატორი. ეს ოპერატორები ამოწმებენ სტეკს, არის თუ არა იქ მათი ოპერანდები. ოპერატორი შლის მათ სტეკიდან და მათ მაგივრად შესრულებული მოქმედების შედეგს ათავსებს. მაგალითად, ოპერატორი add (შეკრება), სტეკიდან შლის ორ ზედა ციფრს და ტოვებს სტეკში მათ ჯამს. ანალოგიურად მუშაობს

ოპერატორი sub, რომელიც ითვლის სხვაობას სტეკის ზედა ციფრსა და მის მომდევნოს შორის.

სხვა არითმეტიკული ოპერატორები:

div გაყოფა. მეორე რიცხვი იყოფა სტეკის სათავეში არსებულ ციფრზე.

13 8 div ==> 1.625

idiv მთელრიცხვა გაყოფა

25 3 idiv ==> 8

mod მეორე რიცხვი იყოფა სტეკის სათავეში არსებულ ციფრზე, გაყოფის ნაშთის დამახსოვრებით.

7 12 mod ==> 5

mod და idiv ოპერატორთა ოპერანდები, მთელი რიცხვები უნდა იყოს.

mul ამრავლებს სტეკის სათავეში მყოფ ორ რიცხვს და მათ მაგივრად ათავსებს ნამრავლს.

8 9 mul ==> 72

neg ცვლის სტეკის სათავეში მოთავსებული რიცხვის ნიშანს.

-37 neg ==> 37

არითმეტიკული გამოსახულებების ჩაწერა.

გამოსახულება 5+(8:2) PostScript ენაზე, შეიძლება წარმოვიდგინოთ რამოდენიმე ხერხით:

82 div 5 add ან 5 8 2 div add

გამოსახულება 9-(4*7), რომელიც შედარებით რთული შემთხვევაა, შეიძლება ჩავწეროთ მინიმუმ ორი ხერხით:

9 4 7 mul sub ან 4 7 mul 9 exch sub

მეორე ვარიანტში, გამოყენებულია ახალი ოპერატორი exch. ეს ოპერატორი სტეკის სათავეში მოთავსებულ ორ ზედა ციფრს ადგილებს უცვლის. exch ოპერატორის გამოყენების აუცილებლობა, გამოწვეულია sub გამოკლების ოპერატორის მოქმედების წესით, ის სათავეში მოთავსებულ რიცხვს აკლებს მის მომდევნო რიცხვს, რასაც exch ოპერატორის გარეშე მოქმედებათა არასწორ თანმიმდევრობასთან მივყავართ.

მოქმედებები სტეკზე.

ოპერატორთა ჯგუფი, რომლის ერთ-ერთი წარმომადგენელია exch სტეკში უმატებს, აკლებს და ცვლის ელემნტთა თანმიმდევრობას.

clear a1 a2 clear

სტეკის გაწმენდის ოპერატორია და სტეკის ყველა ელემენტს შლის 11 6 17 clear ==> -

count a1...an count ==> a1...an n

ითვლის სტეკში ელემენტთა რაოდენობას

dup a1 dup ==> a1 a1

სტეკის ზედა ელემენტის დუბლირებას ახდენს

8 dup ==> 8 8

pop a1 pop

ეს ოპერატორი შლის სტეკის ზედა ელემენტს

31 4 pop ==> 31

roll

ასრულებს სტეკის ელემენტთა ბრუნვას. სტეკიდან ის იღებს ორ რიცხვს. ზედა გვიჩვენებს თუ რამდენჯერ და რა მიმართულებით უნდა მობრუნდეს სტეკის ელემენტი, ხოლო მეორე – რამდენი ელემენტი უნდა მოვაბრუნოთ.

7 8 9 3 1 roll ==> 9 7 8
7 8 9 3 - 1 roll ==> 8 9 7

copy a1...an n copy ==> a1...an a1...an

ეს ოპერატორი სტეკის ზედა n ელემენტის კოპირებას ახდენს.

PostScript ენის *გრაფიკული ნაწილი.*

PostScript ენა, ძირითადად გრაფიკული გამოსახულების მისაღებად დამუშავდა, ამიტომ ოპერატორთა დიდი რაოდენობა გრაფიკას ემსახურება. PostScript–ში ხატვა იდეალურ ზედაპირზე, ანუ მიმდინარე ფურცელზე, გზის კონსტრუირებით იწყება. გზა – წრფისა და მრუდების ნაკრებია, რომელიც ან შესავსებ არეს განსაზღვრავს, ან იმ ტრაექტორიას, რომელიც მიმდინარე ფურცელზე უნდა დაიხატოს. გზის კონსტრუირებისას აუცილებელად უნდა განისაზღვროს მასზე ჩასატარებელი მოქმედებები: შეგვიძლია დავხატოთ მოცემული სისქის წირი ან შევავსოთ იგი, რათა შევქმნათ უწყვეტი გამოსახულება.

მიმდინარე ფურცლის შევსების შემდეგ, შეგვიძლია უკვე მისი ფიზიკურ ფურცელზე დაბეჭდვა. მაგალითად, დავხატოთ ვერტიკალური წირი, რომლის სიგრძეა 5 დუიმი. ეს შესრულდება შემდეგი პროგრამით:

```
newpath
144 72 moveto
144 432 lineto
stroke
showpage
```

ოპერატორი newpath კითხულობს მიმდინარე ფურცელს და აცხადებს, რომ დაიწყო ახალი ფურცლის ხატვა. გზის კონსტრუირება იწყება წარმოსახვითი კალმის მოცემულ წერტილში გადატანით. ეს კალამი გადაადგილებისას, ფურცელზე კვალს არ ტოვებს. კალმის მდგომარეობა ყოველი კონკრეტული მომენტისათვის განიმარტება, როგორც მიმდინარე წერტილს მიმდინარე გზა.

ოპერატორ moveto-ს კალამი გადააქვს წერტილში, რომლის კოორდინატებიც მისი ოპერანდებითაა მოცემული. სტეკიდან ის ორ რიცხვს იღებს და განიხილავს მათ, როგორც მიმდინარე წერტილის x და y კოორდინატს. კოორდინატთა სისტემის სათავე PostScript-ში, ფურცლის მარცხენა ქვედა კუთხეში მდებარეობს. x კოორდინატი იზრდება მარჯვნივ მოძრაობისას, ხოლო y კი – ზემოთ მოძრაობისას. ამ სისტემაში, სიგრძის ერთეული 1/72 დუიმის ტოლია. ამგვარად, ოპერატორი moveto მიმდინარე წერტილს, ორი დუიმით მარჯვნივ (144/72) და ერთი დუიმით ზემოთ (72/72) გადაადგილებს.

ოპერატორი lineto სეგმენტს უმატებს მიმდინარე გზას, ანუ ავლებს წირს მიმდინარე წერტილსა და იმ წერტილს შორის, რომლის კოორდინატებიც ოპერანდებითაა მოცემული. კონკრეტული მაგალითისთვის, ეს რიცხვებია 144 და 432. წერტილი, რომელიც lineto-ს ოპერანდია,

მიმდინარე წერტილი ხდება. სინამდვილეში, lineto არაფერს არ ხატავს მიმდინარე ფურცელზე. ის უბრალოდ წირის სეგმენტს უმატებს მიმდინარე გზას. მოგვიანებით, ამ წირის დახაზვა შესაძლებელია, მაგრამ ეს ავტომატურად არ ხდება.

ოპერატორი stroke უზრუნველყოფს კონსტრუირებული გზის ხატვას მიმდინარე ფურცელზე. შესაბამისად ეს გზა ხილვადი ხდება.

და ბოლოს, ოპერატორი showpage ბეჭდავს მიმდინარე ფურცელს (აგზავნის ბეჭდვაზე).

ამგვარად, შეიძლება გამოვყოთ აგების სამი ეტაპი:

1. გზის კონსტრუირება (newpath, moveto, lineto);
2. მიმდინარე ფურცელზე მისი გადატანა (stroke);
3. მიმდინარე ფურცლის გამოტანა (showpage).

PostScript-ში, ისევე როგორც სხვა ენებში, გადაადგილება მიმდინარე წერტილის მიმართ, შეიძლება იყოს მოცემული არა აბსოლუტურ კოორდინატებში, არამედ ნაზრდებში. ამას ემსახურება ოპერატორი rmoveto და rlineto. ზემოთ განხილული მაგალითი, შეიძლება ასე ჩავწეროთ:

```
newpath
144 72 rmoveto
0 360 rlineto
stroke
showpage
```

განვიხილოთ პროგრამა, რომელიც დახატავს კვადრატს, მისი გვერდები ერთი დუიმის ტოლია, ხოლო ცენტრი ფურცლის ცენტრშია მოთავსებული:

```
newpath
200 300 moveto
0 72 rlineto
0 -72 rlineto
-72 0 rlineto
5 setlinewidth
stroke showpage
```

ნაზრდის გამოყენება საშუალებას გვაძლევს კვადრატი ფურცლის ნებისმიერ ადგილას მოვათავსოთ, და ამისთვის მხოლოდ ერთი სტრიქონის შესწორება გვიწევს. ამ პროგრამაში ერთი ახალი ოპერატორია – setlinewidth, რომელის საშუალებითაც ვადგენთ წირის სიგანეს. ზემოთ მოყვანილ მაგალითში ის 5/72 დუიმის ტოლია. აღნიშნული ოპერატორი, მიმდინარე ფურცელზე მოთავსებულ ყველა ხაზზე მოქმედებს, მანამდე, სანამ მეორე setlinewidth ოპერატორი არ შეხვდება.

კვადრატის მარცხენა ქვედა კუთხეში ამონაჩიჩქნი ჩნდება, რადგან წირს შესამჩნევი სისქე აქვს. ამ მოვლენის თავიდან ასაცილებლად საჭიროა ახლი ოპერატორის გამოყენება – closepath. მოხაზული არის შევსებას fill ოპერატორი ემსახურება, რომელიც კვადრატში შავ მელანს ასხავს.

```
newpath
200 300 moveto
0 72 rlineto
```



```
72 0 rlineto
0 -72 rlineto
closepath
fill
showpage
```

მიაქციეთ ყურადღება, რომ მიმდინარე ფურცელზე წრფეების დატანის (stroke) ოპერატორის მაგივრად, ამჯერად, fill ოპერატორს ვიძახებთ, რომელიც მოხაზულ არეს რაიმე ფერით შეავსებს. fill ოპერატორი მიმდინარე ტრაექტორიას წმინდავს და ამიტომ მიმდინარე წერტილი განუსაზღვრელი ხდება. ფიგურის შევსებისას, რუხი ფერის დონე setgray ოპერატორის არგუმენტითაა მოცემული, ეს კი რიცხვია, რომელიც 0-დან (შავი) 1-მდე (თეთრი) იცვლება.

```
newpath
200 300 moveto
0 72 rlineto
72 0 rlineto
0 -72 rlineto
.6 setgray
fill
showpage
```

რუხის მოცემული დონე მანამდე მოქმედებს, სანამ არ გამოჩნდება შემდეგი setgray ოპერატორი. თუ setgray არ არის მოცემული, მაშინ არე “თავისთავად” ივსება შავი ფერით.

ყოველი ტიპის პრინტერს ნახევარტონების აგების საკუთარი ხერხი აქვს, ამიტომ ერთი და იგივე PostScript –

ფურცლის სხვადასხვა პრინტერზე ნახევარტონებში გამოტანისას, ისინი ერთმანეთს შეიძლება არ დაემთხვეს. ურთიერთ გადამკვეთი არეების ხატვისას, მათი გადაკვეთის ფერი, მიმდინარე ფურცელზე ბოლოს დადებული ფერით განისაზღვრება.

PostScript ენაში ფერთა ორი მოდელი, HSB (ტონი – გაჯერებულობა – სიმკვეთრე) და RGB (წითელი – მწვანე – ცისფერი) აღიქმება. თვითოეულ მოდელში ფერი სამი რიცხვითი პარამეტრით იქმნება. შედარებით მარტივ მოდელ RGB-ში, ფერი სამი ძირითადი ფერის წითლის, მწვანის და ცისფრის ინტენსივობათა შეხამებით მოიცემა. როგორც უკვე ვიცით, ფერთა ინტენსივობა მოცემულია რიცხვებით, 0-დან 1-მდე დიაპაზონში, სადაც 0-ფერის არ არსებობაა, ხოლო 1 კი მისი მაქსიმალური ინტენსივობა. მოდელ HSB-ში ტონი საკუთრივ ფერით მოიცემა. ის განისაზღვრება მისი მდებარეობით ფერთა წრეზე: 0 – სუფთა წითელი, 120 – -მწვანე, 240 – ლურჯი. დანარჩენი ფერები მიიღება ორ მეზობელ ფერთა შერევით. მაგალითად, 60 – ყვითელი, 180 – ცისფერი, 300 – იასამნისფერი. გაჯერებულობა – მოცემული ფერის ტონის გაჯერებულობაა: 0 - შეესაბამება ფერის არ არსებობას, 1 – მის მაქსიმალურ გაჯერებას. სიმკვეთრე – ფერის საერთო ინტენსივობაა (თეთრი ფერის შემცველობა მოცემულ ფერში): 0 – შეესაბამება შავ ფერს, ხოლო 1 – თეთრს (მაქსიმალურ ინტენსივობას).

ფერი, მოდელების შესაბამისი ოპერატორებით setrgbcolor და sethsbcolor-ითაა მოცემული.

შრიფტებთან მუშაობა.

PostScript enis პოპულარობა, ასევე ტექსტის ბეჭდვაზე გამოტანის შესაძლებლობითაა გამოწვეული. ტექსტური მონაცემები PostScript-Si, string (სტრიქონი) ტიპის ობიექტის სახითაა წარმოდგენილი. სტრიქონი შეიძლება შეიცავდეს მრგვალ ფრჩხილებში მოთავსებულ სიმბოლოთა ნებისმიერ თანმიმდევრობას. სტრიქონი სტეკში შეიძლება მოთავსდეს, როგორც ცვლადი ან დაიბეჭდოს. სტრიქონის მიმდინარე ფურცელზე განთავსებამდე, PostScript – ინტერპრეტატორს აუცილებლად უნდა მიუთითოდ შრიფტის გარნიტური და ზომა, რაც მან ბეჭდვისას უნდა გამოიყენოს.

შრიფტი, სიმბოლოთა ნაკრებია, რომელსაც ერთიანი დიზაინი აქვს. კონკრეტული შრიფტის დიზაინს, გარნიტური ეწოდება. გარნიტურთა ნაკრებს, რომელიც დამუშავებულია ერთობლივი გამოყენებისათვის, გარნიტურთა ოჯახი ეწოდება. შედარებით პოპულარული გარნიტურებია: Times New Roman, Arial, Courier, Journal და სხვა. კონკრეტული PostScript-შრიფტი გარნიტურთა ოჯახის ზოგიერთი მოხაზულობის რეალიზაციაა. PostScript-შრიფტები ვექტორულ კლასს მიეკუთვნება, და შესაბამისად, მასშტაბირებადი შრიფტებია. ვექტორული შრიფტების აღწერის არსებული მეთოდები, შრიფტის ზომების ავტომატურად ცვლის საშუალებას იძლევა, მისი მოხაზულობის მინიმალური დამახინჯებით.

შრიფტის მოცემისათვის უნდა შესრულდეს შემდეგი მოქმედებები:

- ვიპოვოთ შრიფტის აღწერა შრიფტების ლექსიკონში. ეს აღწერა, ყოველი ცალკეული სიმბოლოს კონტურის აგების საშუალებას მოგვცემს;
- შევასრულოთ შრიფტის მასშტაბირება საჭირო ზომამდე. მისი ზომა მოიცემა ტექსტის სტრიქონებს შორის იმ აუცილებელი მინიმალური მანძილით ვერტიკალზე, რომელიც სტრიქონების ერთმანეთზე ზედდებას აგვაცილებს. მაგალითად, შრიფტის სიმაღლე ხშირად 12 პუნქტის (გავიხსენოთ, რომ 1 პუნქტი – 1/72 დუიმს) ტოლია;
- დავაყენოთ მასშტაბირებული შრიფტი, როგორც მიმდინარე შრიფტი, რითიც შემდგომში დაიბეჭდება ტექსტი.

ვნახოთ როგორ მუშაობს ეს ყველაფერი, ამისათვის Helvetica შრიფტით, დავბეჭდოთ სიტყვა PC Week, რომლის ზომა 14 პუნქტია.

```
/Helvetica findfont
14 scalefont
setfont
100 150 moveto
(PC Week) show
showpage
```

ამ ფრაგმენტში გამოყენებულია რამოდენიმე ახალი ოპერატორი. პირველი სტრიქონის შესრულებისას სტეკში თავსდება ლიტერალი (ასო) შრიფტის დასახელებით, ამის შემდეგ findfont ოპერატორის გამოძახება ხდება. ეს ოპერატორი ამ სახელის შრიფტს ეძებს ლექსიკონში, რომლის სახელიცაა FontDictionary, და ამ შრიფტის

შესაბამის ლექსიკონს ათავსებს სტეკში. მოცემული ლექსიკონი შეიცავს აღნიშნული შრიფტის სიმბოლოთა აღწერას, რომლის ზომაც ერთი პუნქტის ტოლია. საჭირო ზომა კი დგინდება scalefont ოპერატორის საშუალებით, რომელიც ამ რიცხვს და შრიფტის ლექსიკონს სტეკიდან იღებს და უკან სტეკში უკვე აბრუნებს საჭირო ზომად მოდიფიცირებულ შრიფტის ლექსიკონს. setfont ოპერატორს, შრიფტის ლექსიკონი სტეკიდან მიმდინარე შრიფტში გადაყავს, რომელიც უკვე ტექსტის ბეჭდვისას იქნება გამოყენებული.

PostScript-ში გრაფიკასა და ტექსტს შორის განსხვავება არ არის. ტექსტის სიმბოლო განიხილება როგორც ერთ-ერთი გრაფიკული ობიექტი, რომელიც მიმდინარე ფურცელზეა განთავსებული. ამიტომ ფურცელზე ტექსტისა და გრაფიკის განთავსება რაიმე სპეციალურ მოქმედებებს არ მოითხოვს.

PostScript-პროგრამა, შეიძლება Word-ის დოკუმენტში იყოს “ჩაშენებული”. საქმე იმაშია, რომ Word-ს ფაილთა საკუთარი ფორმატი აქვს, ხოლო PostScript – პროგრამა კი ჩვეულებრივი ASCII-ტექსტია. ამიტომ ამ მიზნისათვის print-ის ველს იყენებენ. გავიხსენოთ, რომ Word-ში ველს უწოდებენ რედაქტორის სპეციალურ ბრძანებებს, რომლებიც ფიგურულ ფრჩხილებშია მოთავსებული. print ველი განკუთვნილია სიმბოლოთა პრინტერზე გამოსატანად (მაგალითად, პრინტერის პირდაპირი მართვის ბრძანებები, ბრძანებები PCL ენაზე და რასაკვირველია PostSript–პროგრამები). ველის ფორმატია:
{print \ p Size "ტექსტი"}

გასაღები \p - გვიჩვენებს, რომ ველში “ტექსტი”, PostSript –პროგრამა ჩაიწერება. PostScript-ის ოპერატორებს, რომლებიც ამ ველშია მოცემული, შეუძლიათ მხოლოდ ხატვის ფანჯარაში მუშაობა და განისაზღვრებიან Size ატრიბუტით, რომლის მნიშვნელობებიც პირველ ცხრილშია მოცემული.

ცხრილი 1.

Size ატრიბუტის შესაძლო მნიშვნელობები

არგუმენტი	მნიშვნელობები
page	ნახატი მთელ მიმდინარე ფურცელზე
para	იმ აბზაცის ფარგლებში (სიმაღლე არანაკლებ დუიმისა), რომელიც print ველს შეიცავს
pic	ნახატი განთავსდება ველის შემდეგ და იმ აბზაცის ბოლომდე, რომელიც ამ ველს შეიცავს
row	ნახატი ცხრილის მიმდინარე სტრიქონში
cell	ნახატი ცხრილის მიმდინარე ელემენტში

ზემოთ განხილული ოპერატორები PostScript-ში არსებული ოპერატორების მხოლოდ უმნიშვნელო ნაწილს შეადგენს, თუმცა ეს საკმარისია ამ ენის თვალსაჩინო დემონსტრაციისათვის, კერძოდ გრაფიკული გამოსახულებისა და ტექსტის ერთობლივი აგებისას.

8. კომპიუტერული გრაფიკის მათემატიკური საფუძვლები

როგორც უკვე ავღნიშნეთ, ვექტორულ გრაფიკას საფუძვლად უდევს გეომეტრიულ ფიგურათა თვისებების მათემატიკური წარმოდგენა. მათემატიკური აპარატის სპეციფიკა, მის პრაქტიკულ მიმართულობაში მდგომარეობს. კომპიუტერული გრაფიკის მათემატიკური მეთოდები, მხედველობისათვის თვალსაჩინო შედეგების მიღებას უზრუნველყოფს. თუმცა გამოყენებითი მათემატიკური მეთოდების გამოყენება, თეორიული საფუძვლების ცოდნისგან არ გვანთავისუფლებს. იგულისხმება ანალიზური გეომეტრიის თეორიის ელემენტები, ორ- და სამგანზომილებიან სივრცეში. ზოგიერთი პრაქტიკული ამოცანის ამოხსნის აუცილებლობიდან გამომდინარე, ხორციელდება თეორიული კონსტრუქციების ეტაპობრივი აგება. ასეთი ერთგვარი არაფორმალური მიდგომა საშუალებას იძლევა განვიხილოთ ანალიზური გეომეტრია არა უბრალოდ როგორც წრფივი ალგებრის ნაწილი, არამედ როგორც იმ პრაქტიკულ გეომეტრიულ ამოცანათა ამოხსნის მძლავრი მეთოდოლოგია, რომელიც კომპიუტერულ გრაფიკაში გვხვდება.

8.1. ანალიზური გეომეტრიის ელემენტები

ვექტორული გრაფიკის უმარტივესი ობიექტია წირი და წერტილი.

წერტილი სიბრტყეზე მოცემულია ორი რიცხვით (x, y), რომელიც განსაზღვრავს მის მდებარეობას კოორდინატთა სათავის მიმართ.

ცნობილია რომ, ***წრფის განისაზღვრისათვის*** საკმარისია ორი პარამეტრი. ჩვეულებრივ წრფის გრაფიკი აღიწერება განტოლებით y=kx+b. თუ ცნობილია k და b, მოცემულ კოორდინატთა სისტემაში ყოველთვის შეიძლება დავხაზოთ უსასრულო წრფე.

წრფის მონაკვეთის განსაზღვრისთვის უნდა ვიცოდეთ ოთხი პარამეტრი, კოორდინატები (x1, y1) და (x2, y2) – მონაკვეთის დასწყისი და დასასრული.

მეორე რიგის მრუდებს მიეკუთვნებათ პარაბოლა, ჰიპერბოლა, ელიფსი, წრეხაზი და სხვა წირები, რომელთა განტოლება არ შეიცავს მეორე რიგზე უფრო მაღალ ხარისხს. წრფე – მეორე რიგის მრუდების კერძო შემთხვევაა. ამ რიგის მრუდები იმით გამოირჩევა, რომ არა არა აქვთ გადაღუნვის წერტილი. მეორე რიგის მრუდის ზოგადი ფორმულა ასე გამოიყურება:

$x^2+a_1y^2+a_2xy+a_3x+a_4y+a_5=0$.

როგორც ჩანს, ხუთი პარამეტრი სავსებით საკმარისია მეორე რიგის უსასრულო მრუდის აღწერისთვის. მეორე რიგის მრუდის მონაკვეთის აღწერისათვის კი საჭირო იქნება ორი პარამეტრით მეტი.

მესამე რიგის მრუდები შედარებით უფრო რთული მრუდებია, მათი განმასხვავებელი თვისება იმაში მდგომარეობს, რომ მათ შეიძლება ქონდეთ გადაღუნვის წერტილი. $y=x^3$ ფუნქციის გრაფიკს გადაღუნვის წერტილი კოოორდინატთა სათავეში აქვს. მესამე რიგის მრუდები კარგად შეესაბამებიან იმ წირებს, რომელსაც ჩვენ ბუნებაში ვხვდებით, მაგალითად ადამიანის სხეულის წირთა ღუნვა, ამიტომ ვექტორული გრაფიკის ძირითადი

ობიექტი სწორედ ასეთი წირია. მეორე რიგის ყველა მრუდი, მესამე რიგის მრუდის კერძო შემთხვევაა.

მესამე რიგის მრუდის განტოლება ზოგადად, ასე ჩაიწერება:

$x^3+a_1y^3+ a_2x^2y+a_3xy^2+a_4x^2+a_5y^2+a_6xy+a_7x+a_8y+a_9=0$.

როგორც ჩანს, მესამე რიგის მრუდის ჩაწერისათვის საკმარისია ცხრა პარამეტრი. მონაკვეთის აღწერისთვის კი ორი პარამეტრით მეტი.

ბრტყელი მრუდები. მრუდების წარმოდგენა

მრუდი შეიძლება წარმოვიდგინოთ, როგორც წერტილთა ერთობლიობა. თუ წერტილები ახლოს არის განლაგებული ერთმანეთთან, მაშინ მათი მონაკვეთით შეერთება, მრუდის გამოსახულებას მოგვცემს. თუმცა ამგვარი წარმოდგენა ხშირად უხარისხოა, განსაკუთრებით როცა მცირეა სიმრუდის რადიუსი. ამგვარი წარმოდგენის გაუმჯობესება, ამ უბნებზე წერტილთა სიმჭიდროვის გავზრდით შეიძლება და საკმაოდ კარგადაც, მაგრამ მიუხედავად ამისა მრუდის ანალიზური წარმოდგენა ბევრად უპირატესია, რადგან ხასიათდება სიზუსტით, ჩანაწერის კომპაქტურობით და შუალედური წერტილების გამოთვლის სიმარტივით. მრუდის ანალიზური წარმოდგენა საშუალებას იძლევა განვსაზღვროთ მრუდის დახრა და რადიუსი, წერტილოვანი წარმოდგენისას კი ამისთვის საჭიროა რიცხვითი დიფერენცირება, რაც საკმაოდ არაზუსტი პროცედურაა. მაგალითად, შევადაროთ წრეხაზის წერტილოვანი და ანალიზური წარმოდგენა. პირველ შემთხვევაში, როცა მასზე 32 წერტილია განლაგებული,

საჭიროა მეხსიერების დიდი მოცულობა, მეორე შემთხვევაში კი ვიმახსოვრებთ მხოლოდ ცენტრის კოორდინატებს და რადიუსს. მრუდის ანალიზური წარმოდგენისას ნებისმიერი წერტილის მდებარეობა ზუსტად განისაზღვრება, ხოლო წერტილოვანის დროს კი საჭიროა ინტერპოლაცია.

პრაქტიკა გვიჩვენებს, რომ ანალიზურად წარმოდგენილი მრუდი ადვილად გამოისახება ნახაზზე. მოხერხებულია მაშინაც, როცა მოცემული კრიტერიუმით საჭიროა მრუდის ფორმის ხშირი ცვლილება. არსებობს ორგანზომილებიანი მრუდის ანალიზური წარმოდგენის სხვადასხვა მეთოდი.

ხშირად საჭიროა ისეთი მრუდის ანალიზური წარმოდგენა, რომელიც თავდაპირველად წერტილებითაა მოცემული. მათემატიკური თვალსაზრისით ეს ინტერპოლაციის პრობლემაა. იმისათვის, რომ მრუდი გავატაროთ ყველა მოცემულ წერტილზე, უბან-უბან პოლინომურ აპროქსიმაციის მეთოდს იყენებენ. ამისათვის საჭიროა რაიმე ხარისხის პოლინომის კოეფიციენტების განსაზღვრა. მრუდის სახე შუალედურ წერტილებში პოლინომის ხარისხსა და საზღვრის პირობებზეა დამოკიდებული.

თუ წერტილები მოცემულია, როგორც მიახლოებითი მნიშვნელობები, მაგალითად ექსპერიმენტული გაზომვის ან დაკვირვების შედეგი, მაშინ საჭიროა ისეთი მრუდი, რომელიც ამ გაზომვებს სწორად მიმართავს. ზოგადად, მრუდმა შეიძლება არცერთ მოცემულ წერტილზე არ გაიაროს. მაგალითად, თუ გამოვიყენებთ უმცირეს კვადრატთა მეთოდს, მაშინ $y = f(x)$ მრუდი, y ღერძზე

საწყის მონაცემებსა და მიღებულ მრუდს შორის გადახრათა კვადრატების ჯამის მინიმიზაციას მოახდენს. $y = f(x)$ ფუნქციის არჩევა კი, შესასწავლი პროცესის ხასიათიდან გამომდინარეობს.

უმცირეს კვადრატთა მეთოდში იყენებენ პოლინომებს, ხარისხობრივ და ექსპონენციალურ ფუნქციებს: $y = ax^b$, $y = ae^{bx}$ ან, $y = c_1 + c_2x + c_3x^2 + ... + c_{n+1}x^n$ სადაც a, b და c_i _ კონსტანტებია. ამ უცნობი კონსტანტების განსაზღვრისათვის მეთოდი, მრუდის არჩევისგან დამოუკიდებლად, ითხოვს წრფივ ალგებრულ განტოლებათა სისტემის ამოხსნას.

არაპარამეტრული მრუდები. მათემატიკური მრუდი შეიძლება წარმოვიდგინოთ პარამეტრული ან არაპარამეტრული ფორმით. არაპარამეტრული მრუდი შეიძლება მოცემული იყოს ცხადი ან არაცხადი ფუნქციის სახით. ბრტყელი მრუდის ცხადი არაპარამეტრული სახეა:

$$y = f(x)$$

მაგალითად, წრფის განტოლება $y = mx + b$, სადაც x–ის ერთ მნიშვნელობას შეესაბამება y-ის მხოლოდ ერთი მნიშვნელობა. ჩაკეტილი ან მრავალსახა მრუდების, მაგალითად წრეხაზის, ცხადი სახით წარმოდგენა არ შეიძლება. არაცხადი წარმოდგენა კი ასეთია

$$f(x,y) = 0.$$

პარამეტრული მრუდები. მრუდის პარამეტრული წარმოდგენისას მისი ყოველი წერტილის კოორდინატი, ერთი პარამეტრის ფუნქციაა. პარამეტრის მნიშვნელობა

მოცემულია მრუდის წერტილის კოორდინატული ვექტორით. t პარამეტრის მქონე ორგანზომილებიანი მრუდისათვის, წერტილის კოორდინატები ტოლია:

$$x = x(t)$$
$$y = y(t)$$

მაშინ, მრუდის წერტილის ვექტორული წარმოდგენა ასეთი იქნება

$$P(t) = [x(t)\ \ y(t)].$$

ბეზიეს მრუდი. მესამე რიგის მრუდის ხაზვა, შესაბამის განტოლებათა მოცემული კოეფიციენტებით, ნაკლებად საინტერესოა. ამ დამღლელი პროცედურის გასამარტივებლად ვექტორულ რედაქტორებში იყენებენ, მესამე რიგის არა ნებისმიერ მრუდს, არამედ მის განსაკუთრებულ სახეს, რომელსაც ***ბეზიეს მრუდი*** ეწოდება. ბეზიეს მრუდის მონაკვეთი – მესამე რიგის მრუდის მონაკვეთის კერძო შემთხვევაა და აღიწერება არა თერთმეტი კოეფიციენტით, არამედ მხოლოდ რვით, რაც მოსახერხებელს ხდის მათთან მუშაობას.

ბეზიეს მრუდის აგების მეთოდი ემყარება წყვილი მხების გამოყენებას, რომელიც წირის ბოლო წერტილებშია გავლებული. პრაქტიკაში ეს მხები "ბერკეტის" როლს ასრულებს, რომელთა საშუალებითაც წირს ისე ვღუნავთ, როგორც გვჭირდება. წირის ფორმაზე მოქმედებს როგორც მხების დახრის კუთხე, ისე მისი მონაკვეთის სიგრძე. მხების (და მასთან ერთად წირის ფორმის) მართვა ხორციელდება მარკერის მაუსით გადაწევით. ვექტორული გრაფიკის რედაქტორთა უმრავლესობა

მრუდების გამოსახვისა და შენახვისთვის სწორედ ბეზიეს მრუდებს იყენებს.

ბეზიეს მრუდი **(Bezier curves) –** პარამეტრული მრუდია, რომელიც მოცემულია შემდეგი გამოსახულებით

$$\mathbf{B}(t) = \sum_{i=0}^{n} \mathbf{P}_i \mathbf{b}_{i,n}(t), \quad 0 < t < 1$$

სადაც $\mathbf{P}_i$ —საყრდენი წვეროების ვექტორთა კომპონენტ ფუნქციაა, ხოლო $\mathbf{b}_{i,n}(t)$ — ბეზიეს მრუდის ბაზური ფუნქციებია, რომელსაც ასევე ბერნშტეინის პოლინომსაც უწოდებენ.

$$\mathbf{b}_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, სადაც n — პოლინომის ხარისხია, i – საყრდენი წვეროს რიგითი ნომერი

როცა n = 1 მრუდი წარმოადგენს წრფის მონაკვეთს, საყრდენი წერტილები $\mathbf{P}_0$ და $\mathbf{P}_1$ განსაზღვრავენ მის დასაწყისს და ბოლოს. მრუდი მოცემულია განტოლებით:

$$\mathbf{B}(t) = (1-t)\mathbf{P}_0 + t\mathbf{P}_1 \quad t \in [0,1].$$

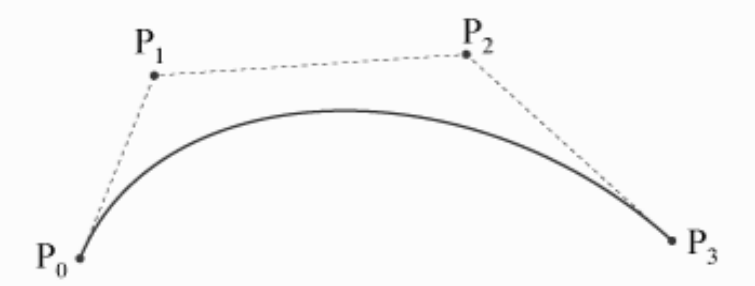
კვადრატული ბეზიეს მრუდი (n = 2) მოცემულია სამი საყრდენი წერტილით: $\mathbf{P}_0$, $\mathbf{P}_1$ და $\mathbf{P}_2$.

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2, \quad t \in [0,1]$$

კუბური ბეზიეს მრუდი (n = 3) აღიწერება შემდეგი განტოლებით:

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3t(1-t)^2\mathbf{P}_1 + 3t^2(1-t)\mathbf{P}_2 + t^3\mathbf{P}_3, \quad t \in [0,1]$$

.



სურ.5 ბეზიეს კუბური მრუდი

ოთხი საყრდენი წერტილი $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}_2$ და $\mathbf{P}_3$, რომელიც მოცემულია ორ- და სამ განზომილებიან სივრცეში, განსაზღვრავს მრუდის ფორმას.

წირი სათავეს იღებს $\mathbf{P}_0$ წერტილში, მიემართება $\mathbf{P}_1$ წერტილისკენ, უახლოვდება $\mathbf{P}_3$ წერტილის $\mathbf{P}_2$ წერტილის მხრიდან და მთავრდება $\mathbf{P}_3$ წერტილში. ამდენად მრუდი არ გაივლის $\mathbf{P}_1$ და $\mathbf{P}_2$ წერტილებს, ისინი ასრულებენ მხოლოდ მიმმართველების როლს. $\mathbf{P}_0$ და $\mathbf{P}_1$ შორის მონაკვეთის სიგრძე კი განსაზღვრავს, როგორ სწრაფად მოუხვევს მრუდი $\mathbf{P}_3$ წერტილისკენ.

თანამედროვე გრაფიკულ სისტემებში მრუდწირული ფორმების წარმოდგენისათვის დიდი გამოიყენება აქვს ***ბეზიეს სპლაინებს*** – რთული ფორმის ხაზების აგებისას

ბეზიეს ცალკეული მრუდები თანმიმდევრობით ერთდება ერთმანეთთან ბეზიეს სპლაინებში.

სპლაინის არსი: შეიძლება ავაგოთ ნებისმიერი ელემენტარული მრუდი თუ ცნობილია ოთხი კოეფიციენტი P0, P1, P2 და P3, რომელიც შეესაბამება ოთხ წერტილს სიბრტყეზე. ამ წერტილთა გადაადგილებით იცვლება მრუდის ფორმა.

სპლაინი (Spline) ვექტორული გრაფიკის ძირითადი ცნებაა – მათემატიკური მრუდია, რომელიც მდორედ აერთებს ცალკეულ წერტილებს. იგულისხმება აგრეგატული ფუნქცია, რომელიც, მის განსაზღვრის არეში, დაყოფის ყოველ ელემენტზე ემთხვევა უფრო მარტივი ბუნების ფუნქციებს.

ერთი ცვლადის კლასიკური სპლაინი აიგება ასე: განსაზღვრის არე იყოფა სასრული რიცხვის მონაკვეთებად, რომელთაგან ყოველზე, სპლაინი ემთხვევა რაიმე ალგებრულ პოლიომს. გამოყენებული პოლინომების ხარისხებს შორის მაქსიმალურს, უწოდებენ სპლაინის ხარისხს. მიღებულ სიგლუვესა და სპლაინის ხარისხს შორის სხვაობას კი სპლაინის დეფექტს. *მაგალითად,* უწყვეტი ტეხილი არის სპლაინი რომლის ხარისხი არი 1 და დეფექტიც არის 1.

წრფივი სურათები – ეს სპლაინებია. სპლაინებზეა აგებული აგრეთვე თანამედროვე შრიფტები. მაგალითად, ბეზიეს კვადრატული მრუდები, როგორც სპლაინების შემადგენელი ელემენტები, გამოიყენებულია True Type შრიფტების სიმბოლოთა ფორმის აღწერისთვის.

უფრო მეტი მნიშვნელობაა აქვს ბეზიეს კუბურ მრუდებს. რაც შეეხება მაღალი ხარისხის მრუდებს, მათი დამუშავებისთვის საჭიროა დიდი მოცულობის გამოთვლები და პრაქტიკული მიზნებისთვის მათ იშვიათად იყენებენ.

8.2. ორგანზომილებიანი გარდაქმნები

8.2.1. წერტილთა გარდაქმნები

წერტილი სიბრტყეზე ორი კოორდინატით განისაზღვრება, ეს კოორდინატები კი 1x2 ზომის [x y] მატრიცის ელემენტებია. სამ განზომილებიან სივრცეში ასეთი მატრიცის ზომაა 1x3 [x y z]. სხვაგვარად რომ ვთქვათ, წერტილი ორგანზომილებიან სივრცეში

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

ვექტორ-კოლონის სახითაა მოცემული, ხოლო სამგანზომილებიან სივრცეში

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

სახით. [x y] სტრიქონს ან კოლონას, ხშირად კოორდინატულ ვექტორს უწოდებენ. ასეთი ვექტორი ფორმირდება მატრიცა-სტრიქონით, ე.ი. წერტილთა სიმრავლით, რომლის ყოველი ელემენტი განსაზღვრავს კოორდინატულ ვექტორს. მოცემული სიმრავლე კომპიუტერში მატრიცის ან რიცხვთა მასივის სახით ინახება. წერტილთა მდებარეობის მართვა შესაბამისი

მატრიცის მანიპულირებით ხდება. წერტილთა შემაერთებელი წირები კი მონაკვეთებს, მრუდებს და სურათებს ქმნიან.

მატრიცის ელემენტი შეიძლება იყოს რიცხვი, ბადე ან განტოლებათა სისტემის კოეფიციენტი. დასაშვებ ოპერაციებს კი ამ ელემენტებზე მატრიცული ალგებრის წესები განსაზღვრავს. ფიზიკის ამოცანათა უმრავლესობა კარგად იხდენს მატრიცულ წარმოდგენას. ასეთი სისტემის მოდელირებისას, ამოცანა ჩვეულებრივ ასე ისმება: მოცემულია [A] და [B] მატრიცა, ვიპოვოთ ისეთი [T] შედეგობრივი მატრიცა, რომ [A][T] = [B]. ამ შემთხვევაში ამონახსნი არის [T] = $[A]^{-1}$ [B] მატრიცა, სადაც $[A]^{-1}$, კვადრატული [A] მატრიცის შებრუნებული მატრიცაა.

ამავდროულად [T] მატრიცა შეიძლება ინტერპრეტირებული იყოს როგორც გეომეტრიული ოპერატორი. მაშინ, [A] მატრიცის იმ წერტილთა გეომეტრიული გარდაქმნა, რომლებიც მდგომარეობის ვექტორითაა მოცემული, მატრიცების გამრავლებით ხდება. ვთქვათ ცნობილია [A] და [T] მატრიცა და საჭიროა [B] მატრიცის ელემენტების განსაზღვრა. მანქანურ გრაფიკაში მათემატიკურ გარდაქმნებს სწორედ, [T]-ს გეომეტრიულ ოპერატორად წარმოდგენა უდევს საფუძვლად.

განვიხილოთ, P წერტილის კოორდინატთა [x y] მატრიცის და ზოგადი გარდაქმნის 2X2 ზომის მატრიცის გადამრავლების შედეგი:

$$[X][T] = [x\ \ y]\begin{vmatrix} a & b \\ c & d \end{vmatrix} [(ax + cy)\ (bx + dy)]$$

მოცემული ჩანაწერი ნიშნავს, რომ P წერტილის x და y საწყისი კოორდინატი გარდაიქმნება x* და y*-ში, სადაც x* = ax + cy, ხოლო y* = bx + dy. ინტერესს წარმოადგენს x*, y* მნიშვნელობები ანუ P წერტილის გარდაქნის შედეგად მიღებული კოორდინატები. განვიხილოთ ზოგიერთი კერძო შემთხვევა.

როცა a = d = 1 და c = b = 0, გარდაქმნას ერთეულოვან მატრიცამდე მივყავართ

$$[X][T] = [x\ \ y]\ \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}\ [x\ \ y] = [x^*\ \ y^*]$$

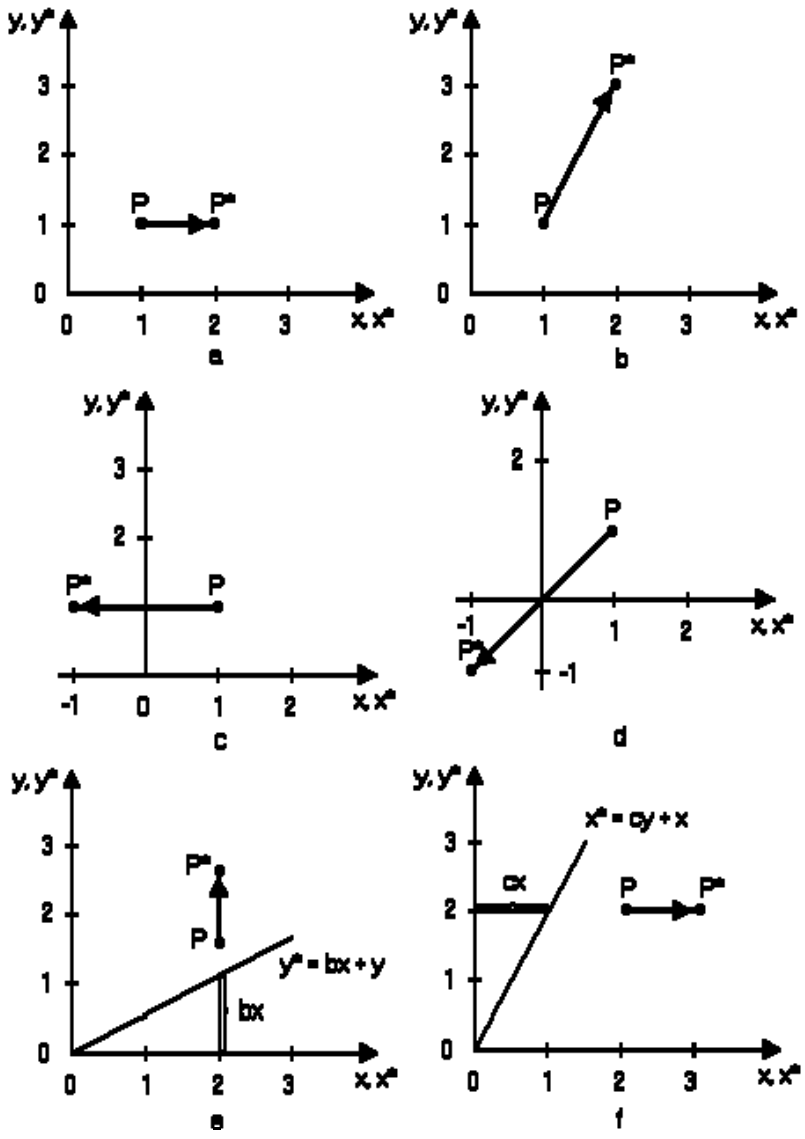
და P წერტილის კოორდინატები უცვლელი რჩება, ვინაიდან წრფივ ალგებრაში ერთეულოვან მატრიცაზე გამრავლება, ჩვეულებრივ ალგებრაში 1-ზე გამრავლების ექვივალენტურია.

როცა d = 1, b = c = 0

$$[X][T] = [x\ \ y]\begin{vmatrix} a & 0 \\ 0 & 1 \end{vmatrix}\ [ax\ \ y] = [x^*\ \ y^*]$$

სადაც x* = ax. ამ გარდაქმნის შედეგი x კოორდინატის მასშტაბირებაა. ასეთი გარდაქმნის ეფექტი ნაჩვენებია სურ. 6.a-ზე.

განვიხილოთ კიდევ ერთი შემთხვევა b = c = 0, ე.ი.



სურ. 6. წერტილთა გარდაქმნები

$$[X][T] = [x\ \ y]\begin{vmatrix} a & 0 \\ 0 & d \end{vmatrix} [ax\ \ yd] = [x^*\ \ y^*]$$

მოცემული გარდაქმნა P ვექტორის, ორივე x , y კოორდინატის ცვლილებას იწვევს (სურ.6.b). თუ a <> d, მაშინ კოორდინატები განსხვავებულად მასშტაბირდება. როცა a = d > 1 ხდება P ვექტორის გაჭიმვა ან კოორდინატთა მასშტაბირება. თუ 0 < a = d < 1, მაშინ ადგილი აქვს შეკუმშვას.

თუ a ან d მნიშვნელობა უარყოფითია, მაშინ ვექტორი კოორდინატთა ღერძის ან სიბრტყის მიმართ აირეკლება. განვიხილოთ შემთხვევა, როცა b= c = 0, d = 1 და a = -1, მაშინ

$$[X][T] = [x\ \ y]\begin{vmatrix} -1 & 0 \\ 0 & 1 \end{vmatrix} [-x\ \ y] = [x^*\ \ y^*]$$

გარდაქმნის შედეგად მივიღებთ y ღერძის მიმართ სიმეტრიულ ანარეკლს (სურ.6.c). ხოლო, თუ b = c = 0, a = 1, d = -1, მაშინ სიმეტრიული არეკვლა x ღერძის მიმართ მოხდება. როცა b = c = 0, a = d < 0, არეკვლა კოორდინატთა სათავის მიმართ ხდება, ეს ნაჩვენებია სურ. 6.d-ზე, სადაც a = -1, d = 1. შევნიშნოთ, რომ ორივე ოპერაცია – არეკვლა და მასშტაბირება, დამოკიდებულია მხოლოდ გარდაქმნის მატრიცის დიაგონალურ წევრებზე.

ახლა განვიხილოთ არადიაგონალური წევრების შემთხვევა. თავდაპირველად ავიღოთ მნიშვნელობები a = d = 1, c = 0, მაშინ

$$[X][T] = [x\ \ y]\begin{vmatrix} 1 & b \\ 0 & 1 \end{vmatrix} [x\ \ (bx + y)] = [x^*\ \ y^*]$$

შევნიშნოთ, რომ P წერტილის x კოორდინატი უცვლელი რჩება, მაშინ როცა y კოორდინატი წრფივად დამოკიდებულია საწყის კოორდინატებზე. მოცემულ გარდაქმნას ძვრა ეწოდება (სურ.6.e). იმ შემთხვევაშიც, როცა a = d = 1, b = 0, გარდაქმნას ანალოგიურად ძვრამდე მივყავართ, y კოორდინატის პროპორციულად (სურ.6.f). ამგვარად, მატრიცის არადიაგონალური წევრების გარდაქმნისას, ვღებულობთ P წერტილის ვექტორის კოორდინატთა ძვრის ეფექტს.

და ბოლოს, განვიხილოთ ზოგადი გარდაქმნა, როცა საწყისი ვექტორი კოორდინატთა სათავეშია მოთავსებული, ე.ი.

$$[X][T] = [x\ \ y]\begin{vmatrix} a & b \\ c & d \end{vmatrix} [(ax + cy)\ (bx + dy)]$$

ან თვით კოორდინატთა სათავის შემთხვევა,

$$[X][T] = [0\ \ 0]\begin{vmatrix} a & b \\ c & d \end{vmatrix} [0\ \ 0] = [x^*\ \ y^*]$$

აქედან ჩანს, რომ კოორდინატთა სათავე ინვარიანტულია (უცვლელია) ზოგადი სახის გარდაქმნის მიმართ. ეს შეზღუდვა იხსნება ერთგვაროვანი კოორდინატების გამოყენებისას.

8.2.2. პარალელური ხაზების გარდაქმნა

წრფე შეიძლება ორი ვექტორით განვსაზღვროთ, რომელიც წრფის ბოლო წერტილთა კოორდინატებითაა მოცემული. ამ წერტილების შემაერთებელი წირი და მისი მიმართულება, შეიძლება ვექტორების მდებარეობის მიხედვით იცვლებოდეს.

ორი პარალელური წირის, 2X2 მატრიცით გარდაქმნის შედეგი, ისევ ორი პარალელური წირია. ამის დასამტკიცებლად, განვიხილოთ $[A] = [x_1\ y_1]$, $[B] = [x_2\ y_2]$ წერტილებს შორის გავლებული და მისი პარალელური E და F წერტილებზე გამავალი წირი. და ვაჩვენოთ, რომ ნებისმიერი გარდაქმნისას ეს წირები პარალელურობას ინარჩუნებს. რადგანაც, AB, EF და A*B* და E*F* პარალელურია, ამიტომ AB და EF ხაზების დახრის კუთხე განისაზღვრება შემდეგი სახით:

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

გარდავქმნათ AB-ს ბოლო წერტილები, რისთვისაც ვიყენებთ 2X2 ზომის ზოგადი გარდაქმნის მატრიცას.

$$\begin{vmatrix} A \\ B \end{vmatrix}[T] = \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix}\begin{vmatrix} a & b \\ c & d \end{vmatrix} = \begin{vmatrix} ax_1 + cy_1 & bx_1 + dy_1 \\ ax_2 + cy_2 & bx_2 + dy_2 \end{vmatrix} = \begin{vmatrix} x_1^* & y_1^* \\ x_2^* & y_2^* \end{vmatrix} = \begin{vmatrix} A^* \\ B^* \end{vmatrix}$$

A*B* წრფის დახრა განისაზღვრება შემდეგნაირად:

$$m^* = \frac{(bx_2 + dy_2) - (bx_1 + dy_1)}{(ax_2 + cy_2) - (ax_1 + cy_1)} = \frac{b(x_2 - x_1) + d(y_2 - y_1)}{a(x_2 - x_1) + c(y_2 - y_1)}$$

ან

$$m^* = \frac{b + d\dfrac{(y_2 - y_1)}{(x_2 - x_1)}}{a + c\dfrac{(y_2 - y_1)}{(x_2 - x_1)}} = \frac{b + dm}{a + cm}$$

რადგანაც m*დახრა არ არის დამოკიდებული x_1, x_2, y_1, y_2 კოორდინატებზე, ხოლო m, a, b, c და d კი EF და AB-სთვის ერთნაერია, ამიტომ m* დახრა E*F* და A*B*-სათვისაც იგივე იქნება. ამრიგად, პარალელური ხაზები გარდაქმნის შემდეგაც ინარჩუნებენ პარალელურობას. ეს ნიშნავს, რომ (2X2) პარალელოგრამის გარდაქმნისას, ის მეორე პარალელოგრამში გარდაიქმნება. ეს ტრივიალური დასკვნები, გარდაქმნის მატრიცათა გამოყენების დიდ შესაძლებლობებზე მეტყველებს, კერძოდ გრაფიკული ეფექტების შექმნისას.

8.2.3. გადამკვეთი ხაზების გარდაქმნა

წყვილ გადამკვეთ წრფეთა გარდაქმნა, (2X2) მატრიცის საშუალებით, ისევ წყვილ გადამკვეთ წრფეებს მოგვცემს. განვიხილოთ ეს შემთხვევა ორი წრფის მაგალითზე, რომლიც შემდეგი განტოლებებითაა მოცემული:

$$y = m_1x + b_1$$
$$y = m_2x + b_2$$

მატრიცულ წარმოდგენაში ეს განტოლებები ასე გამოიყურება:

$$[X][T] = [x \quad y]\begin{vmatrix} -m_1 & -m_2 \\ 1 & 1 \end{vmatrix} = [b_1 \quad b_2]$$

ან

$$[X][M] = [B]$$

თუ ამ განტოლებათა სისტემის ამონახსნი არსებობს, მაშინ ხაზები იკვეთება, წინააღმდეგ შემთხვევაში ისინი პარალელურია. ამონახსნი შეიძლება ვიპოვოთ მატრიცის ინვერსიის გზით. კერძოდ,

$$[X_i] = [x_i \quad y_i] = [B][M]^{-1}$$

[M]-ის შებრუნებულ მატრიცას ასეთი სახეა აქვს:

$$\begin{vmatrix} \frac{1}{m_2 - m_1} & \frac{m_2}{m_2 - m_1} \\ \frac{-1}{m_2 - m_1} & \frac{-m_1}{m_2 - m_1} \end{vmatrix}$$

რადგანაც $[M][M]^{-1} = [E]$, ხოლო [E] კი ერთეულოვანი მატრიცაა, ამიტომ ორი წრფის გადაკვეთის წერტილთა კოორდინატები შეიძლება შემდეგნაერად ვიპოვოთ:

$$[X_i] = [x_i \quad y_i] = [b_1 \quad b_2]\begin{vmatrix} \frac{1}{m_2 - m_1} & \frac{m_2}{m_2 - m_1} \\ \frac{-1}{m_2 - m_1} & \frac{-m_1}{m_2 - m_1} \end{vmatrix}$$

$$[X_i] = [x_i \quad y_i] = \begin{vmatrix} \frac{b_1 - b_2}{m_2 - m_1} & \frac{b_1m_2 - b_2m_1}{m_2 - m_1} \end{vmatrix}$$

თუ ორივე წრფეს, (2X2) ზოგადი გარდაქმნის მატრიცის საშუალებით გარდავქმით,

$$[T] = \begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

მაშინ მათ განტოლებას ასეთი სახე ექნება:

$$y^* = m_1^* x^* + b_1^*$$
$$y^* = m_2^* x^* + b_2^*$$

შესაბამისად

$$m_i^* = \frac{b + dm_i}{a + cm_i}$$

და

$$b_i^* = b_i(d - cm_i^*) = b_i \frac{ad - bc}{a + cm_i} \quad \text{sadac } i = 1, 2.$$

გარდაქმნის შემდეგ წრფეთა გადაკვეთის წერტილს, ისევე ვპოულობთ, როგორც საწყისი წრფეების შემთხვევაში:

$$[X_i^*] = [x_i^* \quad y_i^*] = \begin{vmatrix} \frac{b_1^* - b_2^*}{m_2^* - m_1^*} & \frac{b_1^* m_2^* - b_2^* m_1^*}{m_2^* - m_1^*} \end{vmatrix}$$

თუ წინა სამ გამოსახულებას გამოვიყენებთ, მივიღებთ:

$$[X_i^*] = [x_i^* \quad y_i^*] = \begin{vmatrix} \frac{a(b_1 - b_2) + c(b_1 m_2 - b_2 m_1)}{m_2 - m_1} & \frac{b(b_1 - b_2) + d(b_1 m_2 - b_2 m_1)}{m_2 - m_1} \end{vmatrix}$$

თუ დაუბრუნდებით საწყისი წრფეების გადაკვეთის წერტილს $[x_i \quad y_i]$ და გამოვიყენებთ უკვე მიღებულ გარდაქმნის მატრეიცას, გვექნება:

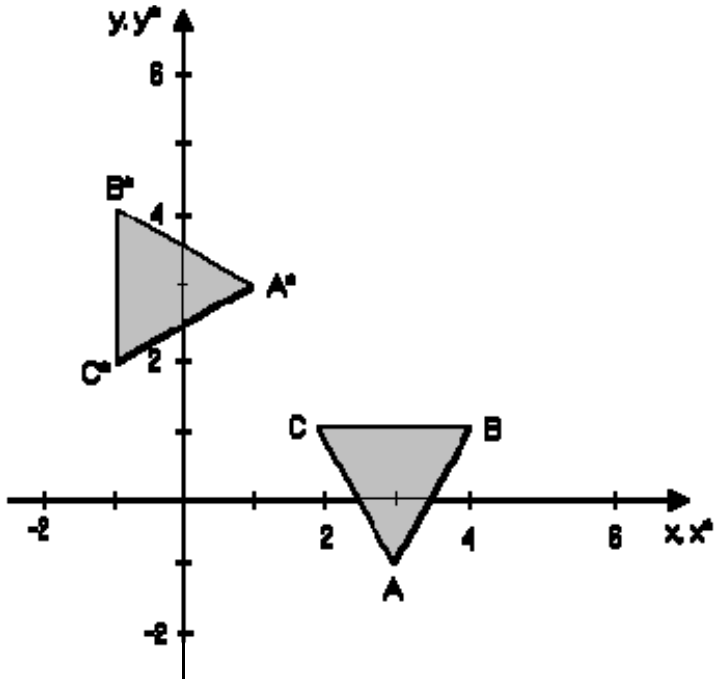
$$[x_i^* \quad y_i^*] = [x_i \quad y_i][T] = \begin{vmatrix} \frac{b_1 - b_2}{m_2 - m_1} & \frac{b_1 m_2 - b_2 m_1}{m_2 - m_1} \end{vmatrix} \begin{vmatrix} a & b \\ c & d \end{vmatrix} =$$

$$= \begin{vmatrix} \frac{a(b_1 - b_2) + c(b_1 m_2 - b_2 m_1)}{m_2 - m_1} & \frac{b(b_1 - b_2) + d(b_1 m_2 - b_2 m_1)}{m_2 - m_1} \end{vmatrix}$$

საწყისი და გარდაქმნილი წირების გადაკვეთის წერტილთა განტოლების შედარება გვიჩვენებს, რომ ისინი ერთნაერია. ამგვარად, გადაკვეთის წერტილი ზუსტად გარდაიქმნება მეორე გადაკვეთის წერტილად.

8.2.4. მობრუნება

განვიხილოთ ABC სამკუთხედი (სურ.7) და 90°-ით მოვაბრუნოთ კოორდინატთა სათავის მიმართ საათის ისრის საწინააღმდეგო მიმართულებით



სურ.7. მობრუნება

$$[T] = \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

გარდაქმნის საშუალებით

თუ გამოვიყენებთ (3X2) მატრიცას, რომელიც შედგენილია სამკუთხედის წვეროთა x და y კოორდინატებისგან, მაშინ შეიძლება ჩავწეროთ

$$\begin{vmatrix} 3 & -1 \\ 4 & 1 \\ 2 & 1 \end{vmatrix} \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = \begin{vmatrix} 3 & -1 \\ 4 & 1 \\ 2 & 1 \end{vmatrix}$$

რაც A*B*C* შედეგობრივი სამკუთხედის კოორდინატებს წარმოადგენს.

კოორდინატთა სათავის მიმართ, 180°-ით მობრუნება შემდეგი გარდაქმნის საშუალებით მიიღწევა

$$[T] = \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$$

ხოლო კოორდინატთა სათავის მიმართ 270°-ით მობრუნება კი

$$[T] = \begin{vmatrix} 0 & -1 \\ 1 & 0 \end{vmatrix}$$

ცხადია, იგივური გარდაქმნის მატრიცა

$$[T] = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}$$

შეესაბამება კოორდინატთა სათავის გარშემო 0° ან 360°-ით შემობრუნებას.

როგორ განვახორციელოთ ნებისმიერი θ კუთხით კოორდინატთა სათავის მიმართ მობრუნება? ამ კითხვაზე პასუხის გასაცემად განვიხილოთ კოორდინატთა სათავიდან P წერტილამდე გავლებული მდგომარეობის ვექტორი (სურ. 8). ვექტორის სიგრძე ავღნიშნოთ r -ით , ხოლო X ღერძსა და ვექტორს შორის კუთხე იყოს φ. მდგომარეობის ვექტორი კოორდინატთა სათავის მიმართ θ კუთხით ბრუნდება და ხვდება P* წერტილში. თუ ჩავწერთ P და P* მდგომარეობის ვექტორებს მივიღებთ:

$$P = [x \quad y] = [r\cos\varphi \quad r\sin\varphi]$$

და

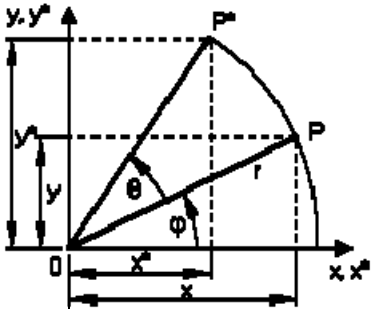
$$P^* = [x^* \quad y^*] = [r\cos(\theta + \varphi) \quad r\sin(\theta + \varphi)].$$

გამოვიყენოთ კუთხეების ჯამის cos-ის ფორმულა და გადავწეროთ გამოსახულება P*–სათვის შემდეგნაირად

$$P^* = [x^* \quad y^*] = [r(\cos\varphi\cos\theta - \sin\varphi\sin\theta) \quad r(\cos\varphi\sin\theta + \sin\varphi\cos\theta)].$$

თუ გამოვიყენებთ x-სა და y –ის განმარტებას, P* შეძლება ასე გადავწეროთ

$$P^* = [x^* \quad y^*] = [x\cos\theta - y\sin\theta \quad x\sin\theta + y\cos\theta].$$



სურ. 8. კოორდინატული ვექტორის მობრუნება

ამგვარად, გარდაქმნილი წერტილის კოორდინატებია

$$x^* = x\cos\theta - y\sin\theta$$
$$y^* = x\sin\theta + y\cos\theta.$$

ხოლო მატრიცული ფორმა კი, ასე გამოიყურება

$$[X^*] = [X][T] = [x^* \quad y^*] = [x \quad y]\begin{vmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{vmatrix}$$

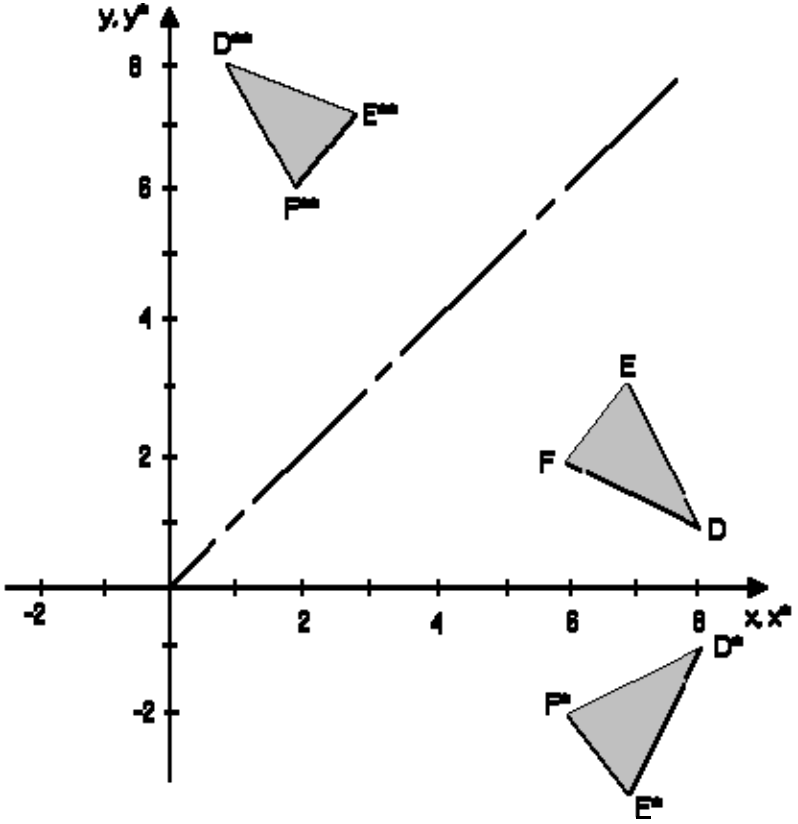
მაშასადამე, კოორდინატთა სათავის მიმართ ნებისმიერი θ კუთხით მობრუნების გარდაქმნა მოცემულია შემდეგი მატრიცით

$$[T] = \begin{vmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{vmatrix}$$

8.2.5. არეკვლა

ორგანზომილებიან სივრცეში, xy სიბრტყეზე სრული მობრუნება, სიბრტყის ნორმალის მიმართ ხორციელდება, არეკვლა კი სამგანზომილებიან სივრცეში, 180°-იანი კუთხით და უკან xy სიბრტყეზე მდებარე ღერძის გასწვრივ იგივე მობრუნებით. სურ.9-ზე მოყვანილია DEF სამკუთხედის სიბრტყეზე არეკვლის ორი მაგალითი. არეკვლა y = 0 (ღერძი X) წრფის მიმართ შემდეგი მატრიცის გამოყენებით მიიღება

$$[T] = \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix}$$



სურ.9. არეკვლა

ამ შემთხვევაში სამკუთხედის ახალი D*E*F* წვეროები შემდეგი გარდაქმნით განისაზღვრება

$$\begin{vmatrix} 8 & 1 \\ 7 & 3 \\ 6 & 2 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} = \begin{vmatrix} 8 & -1 \\ 7 & -3 \\ 6 & -2 \end{vmatrix}$$

ასევე განისაზღვრება არეკვლა y ღერძის მიმართ, როცა x = 0 და მას ასეთი სახე აქვს

$$[T] = \begin{vmatrix} -1 & 0 \\ 0 & 1 \end{vmatrix}$$

y = x წრფის მიმართ არეკვლა კი ხორციელდება

$$[T] = \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}$$

მატრიცის საშუალებით. გარდაქმნის შემდეგ მივიღებთ D**E**F** სამკუთხედის წვეროთა კოორდინატებს

$$\begin{vmatrix} 8 & 1 \\ 7 & 3 \\ 6 & 2 \end{vmatrix} \cdot \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 8 \\ 3 & 7 \\ 2 & 6 \end{vmatrix}$$

არეკვლას x ღერძის მიმართ ანალოგიურად, შემდეგი სახე ექნება

$$[T] = \begin{vmatrix} 0 & -1 \\ -1 & 0 \end{vmatrix}$$

8.2.8. მასშტაბირება

წერტილთა გარდაქმნის შესახებ მსჯელობიდან გამომდინარე მასშტაბირების სიდიდე, საწყისი დიაგონალური მატრიცის ელემენტთა მნიშვნელობებიდან განისაზღვრება. თუ

$$[T] = \begin{vmatrix} 2 & 0 \\ 0 & 2 \end{vmatrix}$$

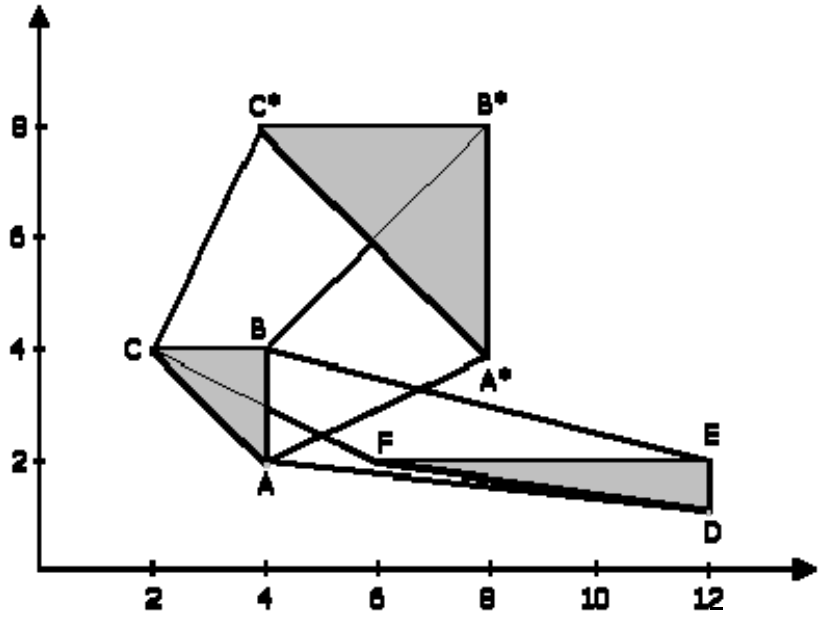
მატრიცას, სამკუთხედის წვეროებზე ზემოქმედების ოპერატორად გამოვიყენებთ, მაშინ ადგილი ექება "ორჯერად" გაფართოებას ან კოორდინატთა სათავის მიმართ თანაბარ მასშტაბირებას. თუ ელემენტთა მნიშვნელობები ტოლი არ არის, მაშინ სამკუთხედი მახინჯდება, რაც კარგად ჩანს მე-10 სურათზე. ABC სამკუთხედი, რომელიც გარდაქმნილია

$$[T] = \begin{vmatrix} 2 & 0 \\ 0 & 2 \end{vmatrix}$$

მატრიცის საშუალებით გადადის პროპორციულად გაზრდილ A*B*C* სამკუთხედში. იგივე სამკუთხედის

$$[T] = \begin{vmatrix} 1/2 & 0 \\ 0 & 3 \end{vmatrix}$$

მატრიცით გარდაქმნისას კი ვღებულობთ DEF დამახინჯებულ სამკუთხედს, რაც მასშტაბირების სხვადასხვა კოეფიციენტებითაა გამოწვეული.



სურ.10. პროპორციული და არაპროპორციული მასსტაბირება

განვიხილოთ ზოგადი შემთხვევა, ანუ გარდაქმნის

$$[T] = \begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

მატრიცა. იმ შემთხვევაში, როცა a = d, b = c = 0 სრულდება პროპორციული მასშტაბირება, ხოლო თუ a <> d, b = c = 0, მაშინ მასშტაბირება არაპროპორციულად მიმდინარეობს. პირველ შემთხვევაში, როცა a = d > 1, გამოსახულება ფართოვდება ანუ დიდდება. ხოლო თუ a = d < 1, მაშინ თანაბრად იკუმშება ანუ მცირდება. არაპროპორციული გაფართოება და შეკუმშვა კი დაკავშირებულია a და d მნიშვნელობებზე, რომლებიც ერთმანეთისგან დამოუკიდებლად, 1-ზე მეტი ან ნაკლები შეიძლება იყოს.

8.2.7. კომბინირებული გარდაქმნები

ფიგურის წვეროთა განმსაზრვრელ კოორდინატულ ვექტორზე მატრიცული ოპერაციების ჩატარებით, შეიძლება ვმართოდ ზედაპირის მდგომარეობა და ფორმა. თუმცა სასურველი ორიენტაციის მისაღებად, შეიძლება დაგვჭირდეს ერთზე მეტი გარდაქმნა. რადგან მატრიცების გამრავლება არ კომუტატურია, ამიტომ მნიშვნელობა აქვს გარდაქმნათა შესრულების თანმიმდევრობას.

მატრიცების გამრავლების არაკომუტატურობის საილუსტრაციოდ, განვიხილოთ [x y] კოორდინატული ვექტორის მობრუნება და არეკვლა. თუ $[T_1]$ მატრიცის საშუალებით, 90° კუთხით მობრუნების შემდეგ, სრულდება არეკვლა $[T_2]$ მატრიცის საშუალებით, y = -x წრფის მიმართ, მაშინ ეს ორი თანმიმდევრული გარდაქმნა მოგვცემს

$$[X'] = [X][T_1] = [x\ \ y]\begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = [-y\ \ x]$$

და შემდეგ

$$[X^*] = [X'][T_2] = [-y\ \ x]\begin{vmatrix} 0 & -1 \\ -1 & 0 \end{vmatrix} = [-x\ \ y]$$

მეორეს მხრივ, თუ შევცვლით თანმიმდევრობას და არეკვლის შემდეგ შევასრულებთ მობრუნებას, მაშინ ასეთ შედეგს მივიღებთ:

$$[X'] = [X][T_2] = [x\ \ y]\begin{vmatrix} 0 & -1 \\ -1 & 0 \end{vmatrix} = [-y\ \ -x]$$

და

$$[X^*] = [X'][T_1] = [-y\ \ -x]\begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = [x\ \ -y]$$

მირებული შედეგები განსხვავებულია, რაც ამტკიცებს რომ მატრიცული გარდაქმნების გამოყენებისას დიდი მნიშვნელობა აქვს მათ თანმიმდევრობას.

8.2.8. ერთეულოვანი კვადრატის გარდაქმნა

აქამდე განიხილებოდა წერტილისა და წრფის ქცევა მარტივი მატრიცული გარდაქმნებისას. თუმცა, მატრიცული გარდაქმნის გამოყენება შეიძლება სიბრტყის ნებისმიერი წერტილისათვის. როგორც უკვე ავღნიშნეთ, ერთადერთი წერტილი, რომელიც ინვარიანტული რჩება მატრიცული გარდაქმნისას, კოორდინატთა სათავეა. სიბრტყის ყველა დანარჩენი წერტილი გარდაიქმნება, და ეს შეიძლება წარმოვიდგინოთ, როგორც კოორდინატთა სისტემის საწყისი სიბრტყის გაჭიმვა და ახალ ფორმაში გადასვლა. ფორმალურად ითვლება, რომ გარდაქმნა, ერთი კოორდინატთა სივრციდან მეორეში გადასვლას იწვევს.

განვიხილოთ კოორდინატთა ბადე, რომელიც xy კოორდინატთა სიბრტყეზე განლაგებული ერთეულოვანი კვადრატებისგან შედგება. კოორდინატთა სათავის მიმართ ერთი და იგივე კუთხით გამავალი ერთეულოვანი

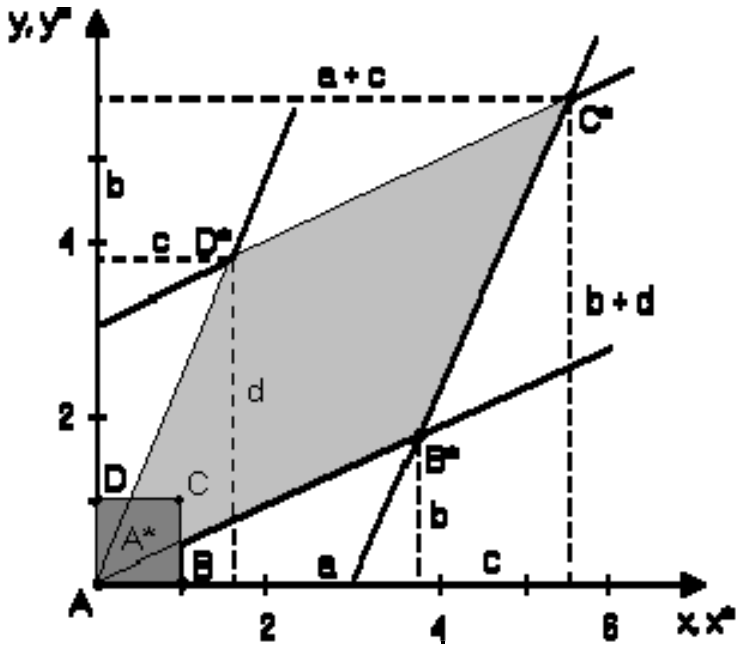
კვადრატის წვეროთა ოთხი კოორდინატული ვექტორი, ასე გამოიყურება:

$$\begin{array}{ll} A & \begin{vmatrix} 0 & 0 \end{vmatrix} \\ B & \begin{vmatrix} 1 & 0 \end{vmatrix} \\ C & \begin{vmatrix} 1 & 1 \end{vmatrix} \\ D & \begin{vmatrix} 0 & 1 \end{vmatrix} \end{array}$$

ასეთი ერთეულოვანი კვადრატი გამოსახულია სურ.11-ზე. თუ მის მიმართ გამოვიყენებთ (2X2) ზოგადი გარდაქმნის მატრიცას, მივიღებთ:

$$\begin{vmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{vmatrix} \begin{vmatrix} a & b \\ c & d \end{vmatrix} = \begin{vmatrix} 0 & 0 \\ a & b \\ a+c & b+d \\ c & d \end{vmatrix}$$

ამ გარდაქმნის შედეგიც სურ.11-ზეა ნაჩვენები.



სურ.11. ერთეულოვანი კვადრატის ზოგადი გარდაქმნა

მიღებული გამოსახულებიდან გამომდინარეობს, რომ კოორდინატთა სათავე გარდაქმნას არ ექვემდებარება, ე.ი. [A] = [A*] = [0 0]. ავღნიშნოთ ასევე რომ, B*-ს კოორდინატები, გარდაქმნის მატრიცის პირველი სტრიქონის ტოლია, ხოლო D*–ს კოორდინატები მეორისა. ამგვარად, გარდაქმნის მატრიცა განსაზღვრულად ითვლება, თუ B* და D* კოორდინატები ([1 0], [0 1] ერთეულოვანი ვექტორების გარდაქმნა) განსაზღვრულია. რადგანა ერთეულოვანი კვადრატის გვერდები საწყის ეტაპზე პარალელურია, ადრე კი ნაჩვენები იყო, რომ პარალელური ხაზები ისევ პარალელურ ხაზებად

გარდაიქმნება, ამიტომ შედეგობრივი ფიგურა პარალელოგრამი იქნება.

მატრიცის a, b, c და d ელემენტების გავლენა კი შეიძლება ცალკე დადგინდეს. როგორც მე-11 სურათიდან ჩანს, b და c ელემენტი, საწყისი კვადრატის y და x მიმართულებით ძვრას იწვევს. ხოლო, a და d ელემენტი მასშტაბირების მამრავლთა როლს თამაშობს. ამგვარად, 2X2 მატრიცა გვაძლევს ძვრისა და მასშტაბირების კომბინაციას.

თუ ისევ მე-11 სურათს დაუბრუნდებით, მაშინ რთული არ იქნება A*B*C*D* პარალელოგრამის ფართობის განსაზღვრაც, რომელიც ასე გამოითვლება:

$$A_p = (a + c)(b + d) - \frac{1}{2}(ab) - \frac{1}{2}(cd) - \frac{c}{2}(b + b + d) - \frac{b}{2}(c + a + c)$$

შედეგად ვღებულობთ

$$A_p = ad - bc = \det\begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

ამგვარად, კვადრატის გარდაქმნით მიღებული ნებისმიერი A_p პარალელოგრამის ფართობი, არის გარდაქმნის მატრიცის დეტერმინანტის ფუნქცია და დაკავშირებულია A_s საწყისი კვადრატის ფართობთან მარტივი შეფარდებით

$$A_p = A_s(ad - bc) = A_s \det [T].$$

ფაქტიურად, რადგან მთლიანი ფიგურის ფართობი ერთეულოვან კვადრატთა ფართობების ჯამის ტოლია, ამიტომ ნებისმიერი გარდაქმნილი A_t ფიგურის ფართობი, დამოკიდებულია საწყისი A_i ფიგურის ფართობზე

$$A_t = A_i(ad - bc).$$

ეს მოსახერხებელი მეთოდია ნებისმიერი ფიგურის ფართობის განსაზღვრისათვის.

8.2.9. გადაადგილება და ერთგვაროვანი კოორდინატები

ზემოთ განვხილულია მთელი რიგი გარდაქმნები, როგორიცაა მობრუნება, არეკვლა, ძვრა მასშტაბირება და სხვა, რომელიც ზოგადი გარდაქმნის (2X2)-ზე მატრიცის გამოყენებით სრულდება. როგორც უკვე აღვნიშნეთ, საწყისი კოორდინატთა სისტემა ინვარიანტულია ყველა ზემოთ ჩამოთვლილი გარდაქმნებისადმი. თუმცა, კოორდინატთა სათავის მდებარეობის შეცვლის, ე.ი. სიბრტყეზე ყოველი წერტილის გარდაქმნის აუცილებლობა ჩნდება. ამის მიიღწევა კი, კორდინატთა სათავის ან სიბრტყის ნებისმიერი სხვა წერტილის გადაადგილების გზითაა შესაძლებელი.

$$x^* = ax + cy + m,$$
$$y^* = bx + dy + n.$$

სამწუხაროთ, ამ გადაადგილების m და n კონსტანტის (2X2)-ზე მატრიცაში შეყვანა შეუძლებელია, რადგან ეს სივრცე ანუ სამგანზომილებიანი არ არის.

ეს სირთულე შეიძლება დავძლიოთ, თუ გამოვიყენებთ ერთგვაროვან კოორდინატებს. არაერთგვაროვანი

კოორდინატული [x y] ვექტორის ერთგვაროვანი კოორდინატები წარმოადგენილია [x' y' h] სახით, სადაც x = x'/h, y = y'/h, ხოლო h – ნამდვილი რიცხვია. შევნიშნოთ, რომ h = 0 განსაკუთრებული შემთხვევაა. ყოველთვის არსებობს ერთგვაროვანი კოორდინატების ერთი ნაკრები რომელსაც ასეთი სახე აქვს [x y 1]. ჩვენ ავირჩიეთ ეს ფორმა, რათა წარმოვადგინოთ [x y] კოორდინატული ვექტორი xy ფიზიკურ სიბრტყეზე. ყველა დანარჩენი ერთგვაროვანი კოორდინატი წარმოდგენილია [hx hy h] სახით. მოცემული კოორდინატები არ ინარჩუნებენ ერთმნიშვნელოვნებას, მაგალითად, [6 4 2], [12 8 4], [3 2 1] კოორდინატები წარმოადგენენ (3,2) ფიზიკურ წერტილს.

ერთგვაროვანი კოორდინატების გარდაქმნის მატრიცის ზომაა 3X3. კერძოდ,

$$[T] = \begin{vmatrix} a & b & 0 \\ c & d & 0 \\ m & n & 1 \end{vmatrix}$$

სადაც a, b, c და d ელემენტთა მოქმედება ზუსტად შეესაბამება (2X2)-ზე მატრიცის ადრე განხილულ მოქმედებებს. m და n ელემენტები კი წარმოადგენენ x და y მიმართულებით გადაადგილების კოეფიციენტებს. გარდაქმნის სრულ ორგანზომილებიან მატრიცას ასეთი სახე აქვს

$$[x^* \;\; y^* \;\; 1] = [x \;\; y \;\; 1] \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{vmatrix} = [x+m \;\; y+n \;\; 1]$$

და ბოლოს ავღნიშნოთ, რომ სიბრტყის ყოველი წერტილი და მათ შორის კოორდინატთა სათავეც x = y = 0, შეიძლება გარდავქნათ.

ერთგვაროვანი კოორდინატების გამოყენება შესაძლებლობას იძლევა ჩავატაროთ გარდაქმნები, როგორიცაა მობრუნება, არეკვლა ძვრა და სხვა, უკვე ნებისმიერი წერტილისა და წრფის მიმართ.

9. კომპიუტერული გრაფიკის პროგრამული ინტერფეისი

კომპიუტერული გრაფიკა შეიძლება განვიხილოთ ასევე, როგორც მეცნიერება აპარატული და პროგრამული უზრუნველყოფის შესახებ, რაც თითქმის ყველა სამეცნიერო და საინჟინრო დისციპლინაში გამოიყენება, იქნება ეს მედიცინა, არქიტექტურა, სარეკლამო ბიზნესი, საგამომცემლო სისტემა თუ სხვადასხვაგვარი ანიმაცია. საბოლოო პროდუქტი ყველა შემთხვევაში გამოსახულებაა, რომელიც შეიძლება იყოს, როგორც უბრალო ტექნიკური ნახაზი ისე ილუსტრაცია, შენობის პროექტი ან კონსტრუქცია, მულტფილმის კადრი ან ანიმაცია. ამ მოძრავ და უძრავ გამოსახულებათა შექმნა, სპეციალური პროგრამული საშუალებებით ხდება.

კომპიუტერული გრაფიკის პროგრამული ინტერფეისი სხვადასხვა ტიპისა და დანიშნულების უამრავ პროგრამას – გრაფიკულ რედაქტორს მოიცავს, რომელიც მომხმარებლისათვის მარტივ, მაგრამ კომპიუტერული გრაფიკის თვალსაზრისით, განვითარებულ და მძლავრ ინსტრუმენტულ საშუალებას წარმოადგენს. ეს კი ნებისმიერი მასალის, როგორც ბეჭდური ისე ეკრანული სახით, მომზადების საშულებას იძლევა.

თანამედროვე გრაფიკული პროგრამები შესაძლებლობათა დიდი რიცხვით ხასიათდება და ეს ჩამონათვალი მუდმივად იზრდება. მაგალითად, პრიმიტივების (მრავალკუთხედი, სპირალი ვარსკვლავი და სხვა) ჩვეულებრივ ატრიბუტებად წარმოდგენა, მრავალფერიანი

გრადიენტული შევსება, მრავალფენიანი ინტერაქტიული ფერადი “დიაპოზიტივები”, ვექტორული გამოსახულების რასტრულში გარდაქმნა და მათი ვექტორული ფაილის ფარგლებში რედაქტირება, რასტრის ტრასირება და ვექტორული სახით წარმოდგენა, დიდ ინტერესს წარმოადგენს აგრეთვე, გარე მოდულები (plug-ins), რომელიც ამდიდრებს ვექტორული გრაფიკისა და ნავიგაციის შესაძლებლობებს და სხვა.

გრაფიკული რედაქტორები, ისევე როგორც ზოგადად გრაფიკა, რასტრული და ვექტორულია და აქაც, პრაქტიკულად არ არსებობს მკვეთრი საზღვარი რასტრულ და ვექტორულ რედაქტორებს შორის. განსაკუთრებული ადგილი უჭირავს 3D-გრაფიკას, რედაქტორებს, რომელთა საშუალებითაც იქმნება სამგანზომილებიანი გამოსახულება. ბოლო წლებში, სამგანზომილებიან სივრცეში, სულ უფრო მეტი საინჟინრო და დიზაინერული სამუშაოები სრულდება.

ისეთი გრაფიკული ობიექტების შექმნა, როგორიცაა ილუსტრაცია, ლოგოტიპი, უზორი, ტექსტური ეფექტები, სქემები და ასევე ანიმაცია, უფრო ეფექტური და მოსახერხებელია ვექტორული გრაფიკის პროგრამათა გამოყენებით, რაც ვექტორული გრაფიკის ძირითად პრინციპებსა და შესაძლებლობებს ეფუძნება. გავიხსენოთ, ობიექტ-ორიენტირებული მიდგომა, ფიგურათა მათემატიკური წარმოდგენა – მათემატიკური მოდელირება, ფრაქტალური გრაფიკის პროგრამული საშუალებები, მასშტაბირება, ფერი, პრიმიტივები და სხვა. და რაც მთავარია, ეს ყველაფერი მუდმივი მოდიფიკაციისა და განვითარების საშუალებას იძლევა.

ვექტორული გრაფიკის რედაქტორი:
Corel Draw
Adobe Illustrator
Macromedia FreeHand
რასტრული გრაფიკის რედაქტორი:
Adobe PhotoShop
Corel Photo Paint
3D-გრაფიკის რედაქტორი:
Autodesk 3ds Max
კომპიუტერული ანიმაციის პროგრამა:
Macromedia Flash
კონსტრუქტორული და არქიტექტურული გრაფიკის რედაქტორი:
AutoCAD
Graphisoft ArchiCAD
Web-გვერდის შესაქმნელი პროგრამა:
DreamWeaver

9.1. ვექტორული გრაფიკის რედაქტორები

ვექტორული გრაფიკის პროგრამებს შორის ყველაზე გავრცელებულია შემდეგი სამი პროგრამა: ***CorelDRAW, Adobe Illustrator*** და ***Macromedia FreeHand***. ვერ ვიტყვით რომ ყველა დანარჩენი პროგრამა ცუდია, ან მხოლოდ CorelDRAW-ა უნაკლო. ყველა პროგრამას აქვს როგორც დადებითი ისე უარყოფითი მხარეები.

Corel Draw

პროგრამა Corel Draw შეიქმნა 90-იანი წლების დასაწყისში. ფირმა Corel-ი ყოველ წელს ანახლებს და ხვეწს ძირითად პროგრამულ პროდუქტს. დღეისათვის შექმნილია CorelDRAW X3 (მე-13 ვერსია). Corel Draw აღიარებულია როგორც ლიდერი. ის მართლაც უნიკალურია, რადგან მარტივი და გასაგებია ნებისმიერი პროფესიისა და ასაკის მომხმარებლისთვის და ამავდროულად ძალიან მძლავრი, უნივერსალური და ინტუიციურია. ეს პროგრამა შეიცავს პროფესიონალურ ფუნქციათა მთელ ნაკრებს, რომელიც მაღალ პროგრამულ დონეზეა შესრულებული და ამიტომ დიდ გამოყენებას პოულობს საგამომცემლო საქმიანობაში.

Corel Draw ილუსტრატიული გრაფიკული პროგრამაა, რომელიც ვექტორული დამუშავების პარალელურად რასტრულ დამუსავებასაც უზრუნველყოფს. ***Corel Draw Graphics Suite X3*** დიზაინისთვის საჭირო პრაქტიკულად ყველა კომპონენტს შეიცავს, როგორიცაა გამოსახულების ტრასირება, შრიფტების რედაქტორი, ტექსტურების მომზადება, შტრიხკოდების შექმნა და გამოსახულებათა უზარმაზარი გალერეა.

- Corel DRAW *X3* _ ვექტორულ გამოსახულებასთან სამუშაო პროგრამა
- Corel PHOTO-PAINT *X3* – რასტრულ გამოსახულებასთან სამუშაო პროგრამა
- Bitstream Fon Navigator _ შრიფტების მართვის პროგრამა
- Microsoft Visual Basic for Applications _ Basic ენის კომპილატორი, რომელიც VBA-ს სცენარების

შექმნის საშუალებას იძლევა CorelDraw-ს დანართებისთვის.

CorelDraw-ს ძირითადი შესაძლებლობები

გრაფიკული გამოსახულების შექმნა და მისი რედაქტირება – ნებისმიერი ფორმის ობიექტის შექმნა, გრაფიკული პრიმიტივები და მათი პარამეტრებით მართვა, Freehand-ის ჯგუფის ინსტრუმენტები: თავისუფალი ხატვა, ბეზიეს მრუდები, მხატვრობა, პულვირიზატორი, ზომების დასმა, მრუდების რედაქტირება და სხვა.

გეომეტრიული ფორმის ობიექტთა რედაქტირება – გეომეტრიული ფორმების შექმნის ინსტრუმენტი, ობიექტის მრუდებში გარდაქმნა, ობიექტის დაყოფა სხვადასხვა ფორმის ინსტრუმენტით როგორიცაა დანა და ლოგიკური ოპერაციების გამოყენება რთული ობიექტის შექმნისათვის.

მაღალი სიზუსტის უზრუნველყოფა – სახაზავი, ბადე, მიმმართველები, ობიექტთა ზუსტი გარდაქმნები, ობიექტების განაწილება და გათანასწორება.

ობიექტთან მუშაობა, ტრანსფორმაცია – კოპირება, დუბლირება, კლონი, მონიშვნა, ობიექტის და დოკუმენტის მასშტაბირება, გადაადგილება, ძვრა, მობრუნება, არეკვლა, დახრა, ობიექტთა თანმიმდევრობის მართვა, დაჯგუფება, განაწილება, გათანაბრება და სხვა.

სპეცეფექტები – ჩრდილის შექმნა, პერსპექტივის დამატება, ფორმის დეფორმაცია, ლინზის ეფექტი, გამჭვირვალობა და სხვა.

ფერის ჩასხმა და კონტური – კონტურთან მუშაობა, ფერის ბუნება, ფერთა მოდელები, გამჭვირვალობა, ობიექტის და კონტურის შეღებვა, ერთი ობიექტის პარამეტრების მინიჭება მეორეზე, პიპეტი და შემვსები, დეკორატიული და მრავალფერიანი გრადიენტული შევსება, უზორი, ტექსტურა, კონტურის და ფერის პარამეტრების მართვა.

ტექსტთან მუშაობა – უბრალო და ფიგურული ტექსტის შექმნა, ფორმატირება, რედაქტირება, ტექსტის მრუდზე განლაგება, ტექსტის გეომეტრიული ფორმის რედაქტირება, ტექსტურ ბლოკებთან მუშაობა და სხვა.

ილუსტრაციასთან მუშაობა – რასტრულ და ვექტორულ გამოსახულებათა იმპორტი, ექსპორტი, რედაქტირება, ვექტორის რასტრში გარდაქმნა, რასტრის ტრასირება, გარფიკული ფორმატები.

ბეჭდვა და Web პუბლიკაცია – დიდი ზომის სურათების ბეჭდვა, ფერთადაყოფა, PS, EPS-ფაილების გენერაცია, მომზადება პოლიგრაფიისა და ელექტრონული პუბლიკაციისათვის და ასევე **Web-**სთვის.

სიახლე პროგრამა CorelDRAW X3-ში

CorelDRAW X3 ვერსიაში დამატებულია ორმოცამდე ახალი ფუნქცია და გარდა ამისა პროგრამა უფრო სწრაფად და სტაბილურად მუშაობს. აღნიშნული სიახლიდან განსაკუთრებულ ყურადღებას იმსახურებს:

- Corel PowerTRACE რომელიც ინტეგრირებულია CorelDRAW X3-ში და საშუალებას იძლევა სწრაფად და მარტივად გარდავქმნათ რასტრული

გამოსახულემა ვექტორულში მისი შემდგომი რედაქტირებისათვის;

- Image Adjustment Lab _ რასტული გამოსახულების როგორც ავტომატური ისე ხელით კორექტირების ფუნქცია;
- ტექსტთან მუშაობის ახალი დახვეწილი ფუნქციები, შეცვლილი ინტერფეისი და გაუმჯობესებული ინსტრუმენტი – Interactive Fit Text to Path;
- სიახლე მრუდების დამუშავებაში – ახალი Shape Tool რეჟიმი ინსტრუმენტ Freehand-ში და ახალი ფუნქცია Reduce nodes;
- კადრირების ახალი ინსტრუმენტი _ Crop Tool, რომელიც ნებისმიერი ფორმატის იმპორტირებული გრაფიკის ზედმეტი უბნების მოცილების საშუალებას იძლევა.

Adobe Illustrator CS

პროგრამა **Adobe Illustrator CS** შექმნილია ცნობილი ფირმა **Adobe Systems inc.**-ის მიერ და ემსახურება გრაფიკული დოკუმენტის პუბლიკაციისათვის მაკეტის შექმნას. სხვა პროგრამებისგან განსხვავებით მას დოკუმენტის მხატვრული გაფორმების ფართო ფუნქციონალური შესაძლებლობები აქვს. ეს პროგრამა **Adobe Creative Suite** გრაფიკული პაკეტის შემადგენლობაში შედის.

Adobe Illustrator -ის ძირითადი შესაძლებლობები

- ვექტორული ობიექტის, სტანდარტული გეომეტრიული და თავისუფალი ფორმის ფიგურათა შექმნა;
- ვექტორული ობიექტის კონტურთა დამუშავება;
- ტექსტური ინფორმაციის დამუშავების მრავალი ფუნქცია;
- ვექტორული ობიექტისა და ტექსტის გაფერადების და კონტურირების დიდი არჩევანი:
  - თანაბარი, გრადიენტული, შაბლონური გაფერადება;
  - თანაბარი, შაბლონური და სხვადასხვა ტიპის ფიგურული კონტური;
- ნებისმიერი ობიექტის რასტრირება;
- გამჭვირვალობისა და ფერთა შერევის რეგულირების შესაძლებლობა;
- გრაფიკული და ტექსტური სტილების გამოყენება, რაც მუშაობას აჩქარებს;
- მხატვრული ეფექტები: დამახინჯება, ჩრდილი, მოზაიკა, ბლიკი და სხვა;
- რასტრულ გამოსახულებაზე სხვადასხვა ფილტრების – ეფექტების გამოყენება;
- ფენეფთან მუშაობა, რაც რთული გამოსახულების შექმნის საშუალებას იძლევა;
- დიაგრამების შექმნა;
- **Web-**გვერდის ფორმირება საწყისი დოკუმენტიდან;

- ანიმაციური ეფექტების შექმნა – დოკუმენტის ფენათა, ეკრანზე თანმიმდევრული ასახვით;

სიახლე პროგრამა *Adobe Illustrator CS*-ში

- Live Trace – ტრასირება, რომელიც ფოტოების, დასკანერებული გამოსახულების და სხვა რასტრული გამოსახულების ვექტორულ ფორმატში სწრაფ და კორექტულ კონვერტაციის საშუალებას იძლევა;
- Live Paint – ინტერაქტიული გაფერადება, ინტუიციურად აფერადებს ნახატს და ავტომატურად პოულობს და აქრობს პრობელებს;
- პალიტრა Control – მართვა, კონტექსტურად მგრძნობიარე პალიტრაა, რომელიც ინსტრუმენტთა პარამეტრების სწრაფი დაყენების საშუალებას იძლევა და გამორიცხავს რამოდენიმე პალიტრის გამოყენებას;
- მომხმარებლის სამუშაო უბანი – ზრდის მუშაობის ეფექტურობას და ეკრანის შემცველეობის ოპტიმიზაციას ახდენს. სამუშაო უბანი შეიძლება დავიმახსოვროთ და გამოვიყენოთ ნებისმიერ დროს როგორც სამუშაო უბნის მზა შაბლონი;
- Photoshop-ის ფენათა კომპოზიციის მხარდაჭერა – პროგრამა Illustrator -იდან Photoshop-ში გახსნილი, ჩასმული ან დაკავშირებული ფაილების ფენათა კომპოზიციების მართვა.

Macromedia FreeHand

კომპანია Macromedia სპეციალიზირებულია მულტიმედიის მაღალმწარმოებლური ინსტრუმენტების დამუშავებაზე. ამ ფირმის პროგრამული პროდუქცია მოიცავს პრაქტიკულად ყველა თანამედროვე მიმართულებას და რაც მთავარია, ისინი კარგად არიან ინტეგრირებული ერთმანეთთან. ყველაზე ცნობილი პროგრამებია _ Macromedia Director, Macromedia Flash, Macromedia Freehand და Macromedia ColdFusion.

Macromedia FreeHand MX _ უნიკალური მრავალფურცლიანი გარემოა, ის საშუალებას იძლევა ვექტორული გრაფიკის გამოყენებით შევქმნათ რთული ილუსტრაცია, მაკეტი, პუბლიკაცია ბეჭდვისთვის და ელექტრონული წარმოდგენისთვის. ამ პროგრამის მძლავრი შესაძლებლობები ეხმარება დიზაინერს განახორციელოს მისი შემოქმედებითი ჩანაფიქრი, ხოლო ილუსტრირების გაფართოებული ფუნქციები, მაკეტირება და პუბლიკაცია კი უზრუნველყოფს სამუშაო პროცესის მოქნილობას.

Macromedia FreeHand MX -is ZiriTadi SesaZleblobebi

- მრავალფერიანი გრადიენტი, ლინზის ეფექტი, გამჭვირვალობა, სამგანზომილებიანი ეფექტები და გარე მოდულების (plug-ins) დამატების შეაძლებლობა;
- ფუნქცია Multiple Attributes, რომელიც ნახატის შექმნისას უზრუნველყოფს შეუზღუდავი რაოდენობის სხვადასხვა სახის დაშტრიხვის, შევსების და სპეცეფექტების გამოყენებას ერთ ვექტორულ ობიექტზე ან ტექსტზე;

- ოპტიმიზირებული სამუშაო პროცესი, რაც საშუალებას იძლევა სწრაფად შეექმნათ გამოსახულება ბეჭდვისთვის, ინტერნეტისა და Macromedia Flash MX პროექტებისთვის;
- სპეციალური სამუშაო პროცესების და ინსტრუმენტების გამოყენებით მრავალფურცლიანი ილუსტრაციების მომზადება, რთული დიზაინის პროექტების შექმნისას;
- რთული ილუსტრაციის და ანიმაციის დამუშავება Macromedia Flash MX –სათვის და ActionScript ბრძანებების გამოყენება;
- ილუსტრაციის შექმნისა და რედაქტირების დაჩქარება ფურცლის შაბლონისა და ფონის, მონაცემთა სიმბოლური ბიბლიოთეკის და ასევე ძებნისა და რედაქტირების ფუნქციათა გამოყენების ხარჯზე;
- მრავალფურცლიანი Web–გვერდების და რთული პროექტების მუშაობის ორგანიზება და ინტერაქტიული პრეზენტაციების მომზადება;

9.2. რასტრული გრაფიკის რედაქტორები

რასტრული გრაფიკის უამრავი რედაქტორი არსებობს (Microsoft Paint, Microsoft Photo Editor, Microsoft Photo DRAW), რომელიც რასტრული გამოსახულების დამუშავების საშუალებას იძლევა, მაგრამ პროფესიონალის მოთხოვნებს მხოლოდ რამოდენიმე მათგანი აკმაყოფილებს, მაგალითად Adobe Photoshop-ი და Corel Photo Paint-ი.

Adobe Photoshop

Adobe Photoshop-ი რასტრული გრაფიკის პროფესიონალური დამუშავების ყველაზე მძლავრი და პოპულარული პროგრამაა. ის ნებისმიერი სირთულის გრაფიკასთან მუშაობის საშუალებას იძლევა. ეს მთელი კომპლექსია, რომელსაც გამოსახულების მოდიფიკაციის მრავალრიცხოვანი შესაძლებლობა, ფილტრებისა და ეფექტების დიდი არჩევანი და ახალი ინსტრუმენტების მიერთების საშუალება გააჩნია. ამ პროგრამის ეფექტურობა განპირობებულია ხატავის, რეტუშირების და კომპოზიციის შესაქმნელი ინსტრუმენტების მრავალფეროვნებით, გამზომი ინსტრუმენტების მოქნილობით, რომელიც სხვადასხვა საზომი ერთეულების კომბინირების საშუალებას იძლევა, ტექსტის დამუშავების ფართო შესაძლებლობებით, როგორიცაა დეფორმაცია და სხვადასხვა ეფექტები, Convert-to-outline ფუნქცია – ტექსტის მასკის სახით გამოყენებისთვის, ორფოგრაფიული შემოწმების მოდული და სხვა.

ამ პროგრამის ბოლო ვერსია PhotoShop CS2, დამატებით შეიცავს პროგრამა ImageReady CS2-ს, რომელიც WEB-გრაფიკის დამუშავების საშუალებას იძლევა (გამოსახულების ოპტიმიზაცია, ანიმირებული gif-ის შექმნა, სურათების დანაწევრება და სხვა), რაც კიდევ უფრო მოსახერხებელს ხდის მას. პროგრამა PhotoShop CS2–ი, Adobe Illustrator CS-ის მსგავსად Adobe Creative Suite-ის შემადგენლობაში შედის.

ეს პროგრამა ახალ სტანდარტს უზრუნველყოფს როგორც ბეჭდვითი გამოცემის, ისე ელექტრონული წიგნის, WEB-

პუბლიკაციისა და უსადენო მოწყობილობისათვის გამოსახულების პროფესიონალური მომზადების სფეროში.

9.3. პროგრამული პაკეტი Adobe Creative Suite 2

განსაკუთრებულ ყურადღებას იმსახურებს პროგრამული პაკეტი Adobe Creative Suite 2, რომელიც კომპანია Adobe–ს სხვადასხვა ბუნების რამოდენიმე გრაფიკულ რედაქტორს აერთიანებს და ქმნის სრულყოფილ, ერთიან დიზაინერულ ინსტრუმენტს. ამ პაკეტის Standard ვერსია შეიცავს შემდეგ პროგრამებს: Photoshop CS2, ImageReady CS2, Illustrator CS2 და InDesign CS2. ვერსია Premium-ს კი დამატებული აქვს პროგრამები GoLive CS2 და Acrobat 7.0 Professional. ეს ორივე ვერსია დამატებით შეიცავს _ Adobe Version Cue-ს ფაილების მართვისთვის და – Adobe Bridge-ს, რომელიც ყველა ამ პროგრამის დამაკავშირებელია.

Adobe Creative Suite 2-ის პროგრამათა ერთობლიობა დიზაინის მძლავრ და უნივერსალურ ინსტრუმენტს ქმნის. ორი მათგანი უკვე განვიხილეთ (Adobe Photoshop და Adobe Illustrator), რაც შეეხება – Adobe InDesign-ს, მას წარმატებით იყენებენ დოკუმენტის დასაკაბადონებლად, Adobe GoLive-ს კი Web-გვერდის შესაქმნელად. როგორც ცნობილია, ელექტრონულ დოკუმენტაციასთან მუშაობის ყველაზე მოსახერხებელი ფორმატია PDF-ი, რომელიც სწორედ პროგრამა Adobe Acrobat-ის საშუალებით იქმნება.

დღეისათვის, ამ პროგრამებს შორის ურთიერთქმედება ძლიერდება. დამატებულია საშუალებები, რომელიც უფრო მეტი საერთო ბრძანებების, ინსტრუმენტების და პალიტრების გამოყენების საშუალებას იძლევა, რაც აადვილებს პროგრამებს შორის გადართვას, ეს კი გაცილებით პროდუქტიულს ხდის სამუშაო პროცესს.

Adobe InDesign CS2

პროგრამა Adobe InDesign CS2-ი ძირითადად განკუთვნილია ბროშურის, წიგნის, მხატვრული ჟურნალის და დიდი ზომის ტექსტის შემცველი სხვადასხვა დოკუმენტის მოსამზადებლად. როცა ასეთი ობიექტის ყველა საჭირო ელემენტი (ტექსტი, ცხრილი, ვექტორული გამოსახულება, ნახატი, ფოტო და სხვა) სხვადასხვა პროგრამაშია შექმნილი, InDesign-is საშუალებით შეიძლება სწრაფად დავაკაბადონოთ და მოვამზადოთ პუბლიკაციისათვის ნებისმიერი სირთულის ფორმა. პროგრამა InDesign-ი ინტეგრირებულია არა მარტო აღნიშნული პაკეტის პროგრამებთან, მისი საშუალებით მარტივად შეიძლება Word-იდან მზა ფორმატირებული ტექსტების, Excel-იდან ცხრილების, PageMaker-იდან ან Quark-იდან აწყობილი გვერდების იმპორტირება.

InDesign-is ძირითადი შესაძლებლობები

დოკუმენტის შექმნა. ფურცლის აწყობა და რედაქტირება, ტექსტური ჩარჩოების შექმნა და რედაქტირება.

ტექსტთან მუშაობა. ტექსტის შექმნა, იმპორტი, რედაქტირება, ეფექტები, კავშირების (Links) შექმნა, შრიფტების მართვა.

ფერი. ფერთა პალიტრა, კონტური, გრადიენტული შევსება, spot-საღებავი, ფერადი ტექსტი.

სხვადასხვა პროგრამიდან გამოსახულების იმპორტი, მათი დაკავშირება (Links) და ამ კავშირების მართვა.

მასტერ-შაბლონებთან მუშაობა. პალიტრა Paragraph Style

ცხრილთან მუშაობა. ცხრილის შექმნა, ფორმატირება და რედაქტირება, მზა ცხრილების შემოტანა, ფორმულებთან მუშაობა.

ვექტორული გამოსახულების შექმნა. მრუდის ხატვა, ფერი, კონტური, კოპირება, ფორმის შეცვლა, არეკვლა.

გრაფიკული ფაილის ფორმატი, გრაფიკის ჩასმის რეჟიმები, Links პალიტრასთან მუშაობა, გრაფიკული ბიბლიოთეკები

ეფექტები. გამჭვირვალობის პარამეტრები, იმპორტირებული გამოსახულების გამჭვირვალობის შეცვლა. ჩრდილის შექმნა.

ინტერაქტიული დოკუმენტის შექმნა. მზა დოკუმენტის ექსპორტი PDF-ში. სარჩევის შექმნა.

პუბლიკაციის მომზადება ბეჭდვისთვის. შრიფტები, PostScript–ფაილის ჩაწერა და ფაილის EPS ფორმატში დამახსოვრება.

ლიტერატურა

1. Петров М.Н., Полочков В.П., Компьютерная графика. 2-ое издание. Питер, 2006.
2. Роджерс Д., Адамс Дж., Математические основы машинной графики. М.: Мир, 2001. — 604 с. ISBN 5-03-002143-4
3. Дик Мак-Клелланд, Photoshop для Windows. Диалектика, 2000, 830с.
4. Федорова А. В. CorelDRAW X3. Экспресс-курс. — СПб.: БХВ-Петербург, 2006. —432 с : ил. ISBN 5-94157-781-8
5. AdobeR Creative Suite 2 [пер. с англ.: Корсаков С, Мингазова Е.]. - М.: Изд-во ТРИУМФ, 2006. — 288 с.: ил. — (Серия «Официальный учебный курс»). - Доп. тит. л. англ. -ISBN 5-89392-145-3.
6. AdobeR PhotoshopR CS : офиц. учеб, курс : [пер. с англ.]. — М. : Изд-во ТРИУМФ, [2004]. - 576 с.: ил. - (Серия "Официальный учебный курс"). —Доп. тит. л. англ. — ISBN 5-89392-091-0, ISBN 0-32M9375-X (амер.)

იბეჭდება ავტორთა მიერ წარმოდგნილი სახით
კომპიუტერული უზრუნველყოფა მ. თუშიშვილის

გადაეცა წარმოებას 26.05.2007 წ. ხელმოწერილია დასაბეჭდად
06.06.2007 წ. ქაღალდის ზომა 60X84 1/16. ბეჭდვა ოფსეტური.
პირობითი ნაბეჭდი თაბახი 5,625. ტირაჟი 100 ეგზ.

საგამომცემლო სახლი “ტექნიკური უნივერსიტეტი”,
თბილისი, კოსტავას 77