Proširenje programskog jezika Logo na 3D i konkretna implementacija u obliku sučelja za 3D animacije Elica

Tomičić, Igor

Master's thesis / Diplomski rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:091014

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-20**

Repository / Repozitorij:

Repository of the Faculty of Science - University of Zagreb

SVEČILIŠTE U ZAGREBU
PRIRODOSLOVNO–MATEMATIČKI FAKULTET
MATEMATIČKI ODSJEK

Igor Tomičić

Proširenje programskog jezika Logo na 3D i konkretna implementacija u obliku sučelja za 3D animacije Elica

Diplomski rad

Voditelj rada:
Goran Igaly

Zagreb, 2018.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

UVOD **1**

1. POVIJEST PROGRAMSKOG JEZIKA LOGO **2**

2. BITNE VERZIJE PROGRAMSKOG JEZIKA LOGO **5**

2.1. FMSLogo 5
2.2. Terrapin Logo 7
2.3. Elica 9
2.3.1. Razvoj Elice 9
2.3.2. Transkript intervjua s prof. Pavelom Boytchevom 11

3. CRTANJE POMOĆU LOGA U HRVATSKIM ŠKOLAMA **17**

3.1. Crtanje geometrijskih likova 18
3.2. Crtanje geometrijskih tijela 25
3.3. Trodimenzionalno crtanje u FMSLogu 30

4. NOVI NAČINI CRTANJA U ELICI **35**

5. APLIKACIJE I ANIMACIJE UNUTAR ELICE **43**

5.1. DALEST Stuffed Toys 43
5.2. DALEST Scissors 45
5.3. Around the World 47
5.4. Solar System 52

LITERATURA **56**

SAŽETAK **58**

SUMMARY **59**

ŽIVOTOPIS **60**

Uvod

Logo se u nastavi informatike koristi već u nižim razredima osnovne škole te se smatra jednostavnim programskim jezikom, prikladnim djeci i učenicima koji se prvi puta susreću s programiranjem. Ovaj diplomski rad će pokazati više mogućnosti Loga vezane za crtanje kornjačom.

Glavni dio rada sastoji se od pet poglavlja: *Povijest programskog jezika Logo*, *Bitne verzije programskog jezika Logo*, *Crtanje pomoću Loga u hrvatskim školama*, *Novi načini crtanja u Elici* i *Aplikacije i animacije unutar Elice*.

U prvom poglavlju, *Povijest programskog jezika Logo*, ću pisati o pozadini i ideji za izradom tog programskog jezika, kao i razvoju Loga tijekom prvih godina postojanja.

U drugom poglavlju, naziva *Bitne verzije programskog jezika Logo*, ću navesti tri verzije Loga bitne za hrvatske korisnike i čitatelje ovog rada te opisati njihov razvoj i osnovne mogućnosti.

U trećem poglavlju, *Crtanje pomoću Loga u hrvatskim školama*, ću opisati kako se u hrvatskim školama koje uče djecu programiranju u Logu crtaju geometrijski likovi i poneka tijela te ću se dotaknuti i naredbi za crtanje u trodimenzionalnom prostoru.

U četvrtom poglavlju naziva *Novi načini crtanja u Elici* ću uvesti nove naredbe za upravljanjem kornjačom, jedinstvene za tu implementaciju programskog jezika Logo, te demonstrirati njihov rad. Također ću opisati naredbe kojom možemo izraditi kratke animacije ili promijeniti kut gledanja na sliku.

Konačno, u petom poglavlju *Aplikacije i animacije unutar Elice* ću predstaviti neke od aplikacija i animacija dostupnih unutar Elice. Konkretno, opisat ću rad aplikacija koje pomažu učenicima prilikom obrade geometrijskih tijela i njihovih mreža te ću predstaviti dvije animacije koje se mogu koristiti u nastavi Geografije.

1. Povijest programskog jezika Logo

Ideja programskog jezika koji bi se koristio tijekom obrazovanja djece nastala je sredinom 1960.-ih godina u privatnoj istraživačkoj tvrtki *Bolt, Beranek and Newman* (BBN). Na projektu su prvenstveno radili Wallace Feurzeig i Cynthia Solomon, djelatnici odjela za istraživanje umjetne inteligencije u BBN-u, te Seymour Papert, koji je prethodno radio s Jeanom Piagetom (tvorcem teorije kognitivnog razvoja djece) u Ženevi, konzultant iz MIT-a.

Slika 1.1: Redom slijeva nadesno: Wallace Feurzeig, Cynthia Solomon i Seymour Papert

Shvativši da su postojeći programski jezici stvoreni prvenstveno za izračune te kao takvi nisu podobni za nenumeričku simboličku manipulaciju, odlučili su stvoriti novi programski jezik dizajniran za poučavanje, sa sljedećim osnovnim zahtjevima:

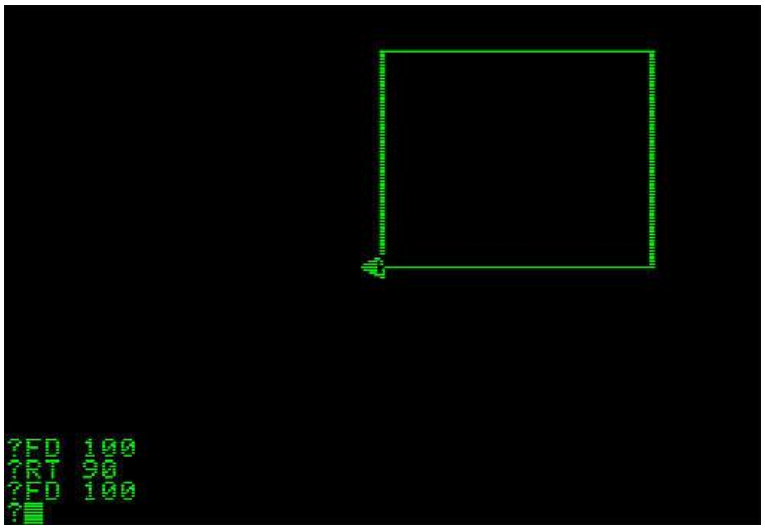
1) Učenici trećeg razreda bi ga trebali moći koristiti za jednostavne probleme bez puno pripreme.
2) Njegova struktura bi trebala sadržavati matematički važne koncepte s minimalno ometanja uslijed programskih konvencija.
3) Trebao bi omogućiti izražavanje nenumeričkih, kao i numeričkih algoritama. Primjeri nenumeričkih problema uključuju stvaranje i razbijanje „tajnih kôdova“, igre riječi poput otkrivanja je li neka riječ palindrom, pronalaženje riječi unutar riječi, pisanje riječi unatrag i sl.

Tijekom 1966. i 1967. stvoren je i razvijen programski jezik *Logo*, čiji naziv dolazi od grčke riječi *logos*, što znači riječ ili misao. Već od 1967. počinje pilot projekt rada s učenicima petog i šestog razreda. Kako u to vrijeme još nije izumljen mikroprocesor, pa tako ni osobna računala, učenici su se spajali putem terminala na pdP-1 računalo unutar BBN-a. Rezultate su dobivali ispisane putem teleprintera.

1969. napravljen je prvi robot „kornjača“, upravljan pomoću programskog jezika Logo. Robot se mogao kretati naprijed i unatrag te okretati ulijevo ili udesno za zadani kut. Osim toga, na njega je bila pričvršćena olovka koju je mogao podignuti ili spustiti kako bi ostavljao trag. Naziv „kornjača“ je dobio zbog pokrova koji je ličio na oklop kornjače.

Slika 1.2: Robot „kornjača“, preuzeto s http://cyberneticzoo.com/wp-content/uploads/22-turtle2-x640.jpg

Po uzoru na robota „kornjaču“, početkom 1970.-ih dodana je grafička „kornjača“ unutar programskog jezika Logo. Osim naredbi identičnih robotu, dodane su naredbe za brisanje ekrana, resetiranje pozicije kornjače, debljinu crte i sl. Upravo je pojava virtualne kornjače popularizirala programski jezik Logo među djecom i osigurala njegov opstanak u školama diljem svijeta sve do današnjice.



Slika 1.3: Grafička „kornjača“, preuzeto s https://theantiroom.files.wordpress.com/2011/04/logo_turtle.jpg

Godine 1970. Seymour Papert na MIT-u unutar Zavoda za umjetnu inteligenciju osniva Laboratorij za Logo koji preuzima vodeću ulogu u razvoju tog programskog jezika.

Sljedeći veliki korak u popularizaciji programskog jezika Logo dogodio se pojavom osobnih računala krajem 1970.-ih, posebice izrazito popularnog i cjenovno prihvatljivog Apple II računala. Pojavom različitih osobnih računala također su se razvijale različite verzije programskog jezika Logo, svaka s određenim karakteristikama.

Značajan utjecaj na Logo u nastavi dogodio se 1980. kada je Seymour Papert objavio knjigu *„Mindstorms: Children, Computers, and Powerful Ideas“*. Knjiga je imala utjecaj na mnoge nastavnike koji su, uz pomoć sve prisutnijih osobnih računala u školama, u većoj mjeri počeli poučavati svoje učenike programskom jeziku Logo, a samim time i algoritamskom načinu razmišljanja.

Logo se nastavio razvijati sve do danas te je i dalje dio nastavnih kurikuluma u mnogim državama, uključujući i Republiku Hrvatsku.

2. Bitne verzije programskog jezika Logo

Od svog nastanka do danas programski jezik Logo pojavio se u raznim izdanjima. U početku je svako novo izdanje donosilo novitete i dodatne mogućnosti, a pojavom osobnih računala ukazala se potreba za posebnim verzijama prilagođenim odgovarajućim operacijskim sustavima. Pojavom Interneta nastale su i web implementacije.

Do danas je nastalo preko 300 različitih verzija i implementacija programskog jezika Logo. U ovom poglavlju ću spomenuti tri verzije bitne za hrvatske korisnike i čitatelje ovog rada.

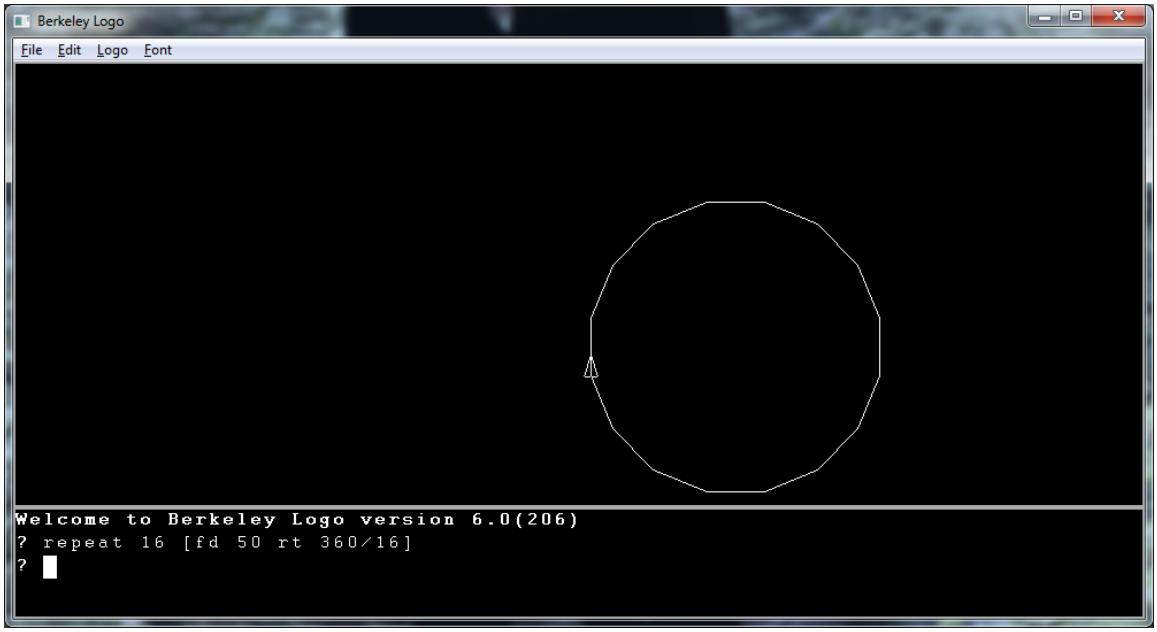
2.1. FMSLogo

Početkom 1990.-ih godina profesor Brian Harvey, predavač na Sveučilištu u Berkeleyju u Kaliforniji (UCB), htio je napisati udžbenike o programskom jeziku Logo za djecu, nastavnike i neprofesionalne programere. Kako je u to vrijeme postojalo mnogo različitih implementacija tog programskog jezika koje su koristile različite sintakse (primjerice, kako bi zbrojili brojeve 2 i 3 u jednoj implementaciji bi upisali očekivani izraz 2+3, no u drugoj bi morali dodati razmake oko znaka za zbrajanje: 2 + 3), a i posjedovale raznolike mogućnosti, profesor Harvey je odlučio objediniti najbolje od svake verzije i napraviti implementaciju koja će imati podržane verzije za Macintosh, Unix, DOS i Windows operacijske sustave. Tako je 1993. nastao Berkeley Logo ili UCBLogo, implementacija dostupna svima – i to besplatno, u vrijeme kad se većina popularnih implementacija trebala kupiti!

UCBLogo smatra se standardom modernog Loga: uključuje riječi, liste i matrice (broj je specijalni slučaj riječi) te koristi dvotočku (`:`) kao pokazivač na sadržaj varijable i navodnike (`"`) kao početak riječi. Primjerice, izraz

```
make "a sum :x 5
```

zbraja sadržaj varijable `x` s brojem 5 i rezultat sprema u varijablu `a`.

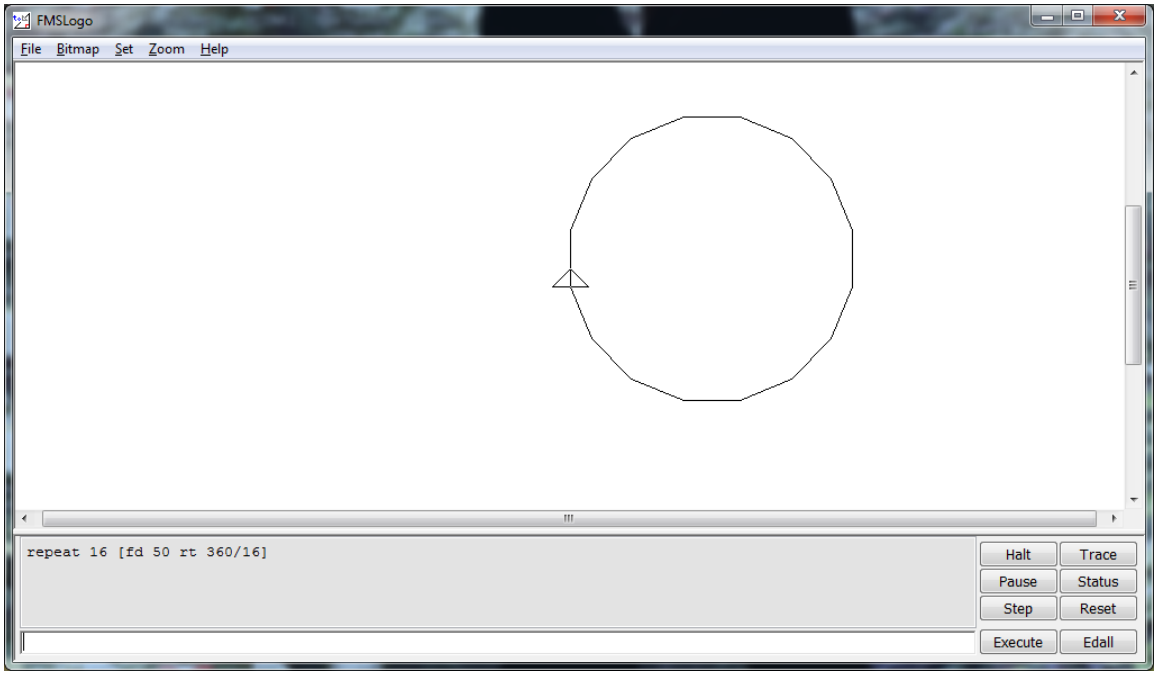


Slika 2.1: UCBLogo – primjer crtanja šesnaesterokuta

Kratko nakon nastanka UCBLoga, George Mills s MIT-a je u razvojnom okruženju Borland C++ napravio poboljšanu verziju UCBLoga – iskoristio je mogućnosti Microsoft Windowsa te dodao višestruke kornjače i 3D grafiku. Kako je ta verzija radila samo na operacijskom sustavu Microsoft Windows, nazvana je MSWLogo. Bitno je napomenuti kako je i ova, poboljšana verzija UCBLoga, i dalje bila besplatna te otvorenog kôda.

U narednih desetak godina, zahvaljujući svojim mogućnostima i dostupnosti, MSWLogo je postao jedna od najpoznatijih implementacija programskog jezika Logo, posebice u školama. Mnogi udžbenici su pisani upravo za MSWLogo te su brojne generacije odrasle upravo na ovoj verziji.

2004. godine David Costanzo je Georgu Millsu predložio nekoliko malih poboljšanja MSWLoga. Kako Mills više nije imao vremena za rad na Logu, Costanzo je 2005. napravio svoju verziju s uključenim poboljšanjima i planovima za daljnje poboljšanje. Kako bi izbjegao legalni konflikt s Microsoftom, za kojeg je tada počeo raditi, Costanzo je morao maknuti MSW iz naziva. Saznavši da su bližnji Georga Millsa oboljeli od multiple skleroze, nova implementacija je nazvana FMSLogo, gdje prefiks FMS znači „Fight Multiple Sclerosis“.



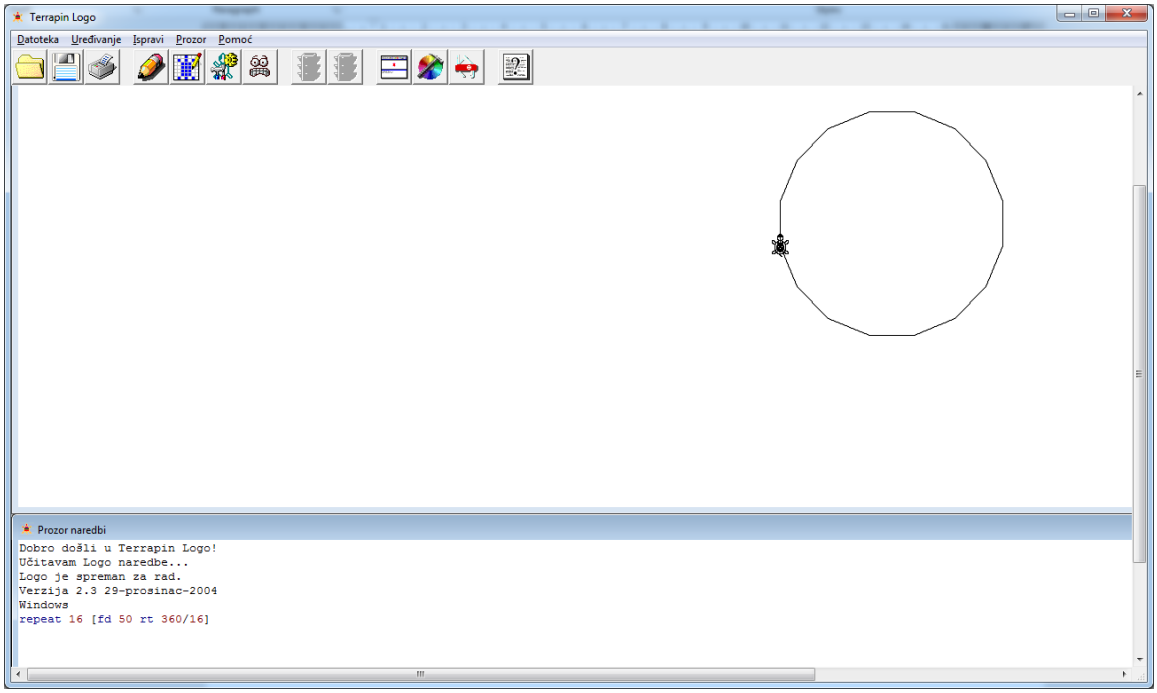
Slika 2.2: FMSLogo – primjer crtanja šesnaesterokuta

FMSLogo se u našim školama koristi i danas te je najraširenija implementacija Loga u Republici Hrvatskoj. Iako je vizualno neatraktivan, njegove mogućnosti, besplatan pristup i lakoća korištenja opravdavaju taj izbor.

U budućnosti FMSLogo bi trebao podržavati Unicode standard, dobiti 64-bitnu podršku, poboljšati funkciju Editora (uređivača), imati verziju za GNU i Linux te dobiti nove mogućnosti poput vektorske grafike, animacija i podrške za pretvaranje teksta u govor.

2.2. Terrapin Logo

U našim školama pojavljuje se i komercijalni Terrapin Logo, preveden na hrvatski jezik. Koriste ga učenici koji rade po udžbenicima nakladnika SysPrint, a sadržaje vezane uz Terrapin Logo je napisala profesorica Ines Kniewald, jedna od naših najvećih stručnjakinja za Logo.



Slika 2.3: Terrapin Logo – primjer crtanja šesnaesterokuta

Terrapin je najstarija tvrtka posvećena komercijalnim izdanjima programskog jezika Logo, a njen naziv dolazi od vrste kornjača. Tvrtka Terrapin postoji od 1979. godine, a Terrapin Logo je napravljen 1980. godine.

Radi se o vizualno najprivlačnijoj verziji Loga, s različito obojanim naredbama, varijablama i vrijednostima, kornjačom koja uistinu izgleda kao kornjača (a ne trokut), pojednostavljenom sintaksom (za dohvat sadržaja varijable `a` dovoljno je napisati `a` umjesto `:a`), pregršt boja i za crtu i za samu kornjaču, stotinama uključenih grafičkih i zvučnih elemenata, spremanjem radne okoline koja će biti identična na drugom računalu, kompatibilnošću među verzijama za različite operacijske sustave itd.

Upravo zbog svog atraktivnog izgleda i pojednostavljene sintakse Terrapin Logo je vjerojatno najprivlačnija verzija učenicima osnovnih škola.

2.3. Elica

Elica je bila besplatna i jedna od najmodernijih implementacija programskog jezika Logo. Osim standardnih Logo mogućnosti, također je omogućavala 3D animacije objekata i podršku za objektno orijentirano programiranje, bez potrebe za uvođenjem nove sintakse. Primarno je bila namijenjena iskusnijim i zahtjevnijim korisnicima koji su već bili dobro upoznati s mogućnostima standardnih Logo implementacija. Kao moderni obrazovni alat za prvo desetljeće ovog tisućljeća, Elica je korištena za stvaranje stotina animiranih virtualnih modela, matematičkih vizualizacija i multimedijalnih sadržaja.

Na žalost, sav rad na ovoj obećavajućoj implementaciji programskog jezika Logo je prestao 2013. godine.

2.3.1. Razvoj Elice

Prva verzija Elice je nastala u periodu od 1999. do 2001. godine. Autor Elice je Pavel Boytchev, izvanredni profesor Fakulteta matematike i informatike Sveučilišta u Sofiji, Bugarska. Naziv Elica prvotno dolazi od akronima za „Educational Logo Interface for Creative Activities“, no prof. Boytchev je kasnije promijenio izvor akronima u po njemu prikladniji „Educational Logo Interface for Computer Animation“.

Slika 2.4: Pavel Boytchev

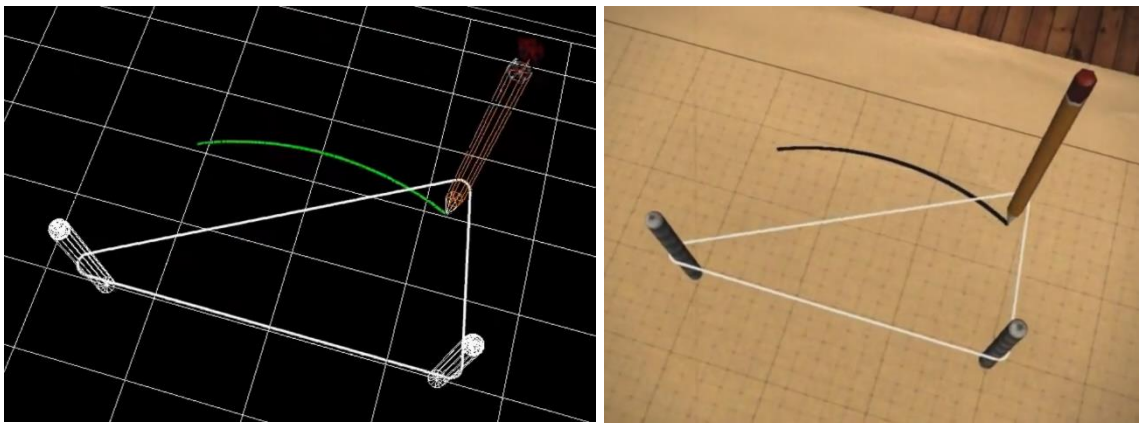
Glavni cilj Elice je bio proširiti programski jezik Logo s objektno orijentiranim programiranjem i 3D grafikom, no bez uvođenja nove sintakse. Iako je projekt u raznim fazama uključivao više ljudi, svo programiranje Elice je radio prof. Boytchev.

Glavni ciljevi tijekom razvoja su bili dodati funkcionalnosti koje će omogućiti brzu izradu edukacijskog softvera i 3D grafiku za obrazovne svrhe. Ciljevi koji nisu ostvareni prije kraja razvoja su bili učiniti Elicu prijenosnom na druge operacijske sustave osim Windowsa, napraviti online verziju Elice itd.

Iako se sâma Elica nije koristila u obrazovanju, koristile su se aplikacije napisane pomoću Elice. Primjerice, prof. Boytchev je napravio nekoliko aplikacija za projekt DALEST (Developing Active Learning Environment for Stereometry), u kojem je sudjelovalo više sveučilišta i drugih obrazovnih ustanova diljem Europe. Nadalje, prof. Boytchev je napravio velik broj manjih aplikacija za osnovnoškolsku i srednjoškolsku geometriju koje su se koristile u nekim europskim školama, a neke su opisane i u bugarskim osnovnoškolskim udžbenicima.

Iako je većina ljudi bila impresionirana mogućnostima Elice, nitko osim profesora Boytcheva nije htio raditi nove aplikacije u njoj. Nadalje, nije bilo zainteresiranih koji bi pružili podršku daljnjem razvoju Elice, bilo financijski ili na druge načine. Nastavno na sve veće obveze profesora Boytcheva i nekoliko konflikata između njega i Logo zajednice, 2013. je stao sav razvoj na Elici te je prof. Boytchev prekinuo sav rad vezan za Logo (programiranje, objavljivanje radova, prezentacije i predavanja).

Srećom, na web stranici http://www.elica.net/ sve verzije Elice su i dalje dostupne za preuzimanje, kao i dio napravljenih aplikacija i animacija.



Slika 2.5: Elica – isječak iz animacije crtanja elipse (lijevo: *wireframe* verzija, desno: verzija s teksturama)

2.3.2. Transkript intervjua s prof. Pavelom Boytchevom

U nastavku prenosim prijevod i originalni transkript na engleskom jeziku kratkog intervjua e-mailom s profesorom Pavelom Boytchevom koji mi je dao značajno detaljniji uvid u razvoj Elice u odnosu na izvore dostupne na Internetu. Intervju je proveden u siječnju 2017. godine.

Slovom I sam označio svoja pitanja, a slovom P odgovore profesora Boytcheva.

Intervju preveden na hrvatski jezik:

I: Što Vas je prvenstveno potaknulo da napravite Elicu i kada je objavljena prva verzija?

P: Napravio sam nekoliko implementacija jezika Logo. Elica je bila jedna od njih. Dakle, to je pomalo nastavak mojih prethodnih pokušaja. Glavni cilj Elice bio je proširiti Logo s objektno orijentiranim programiranjem i trodimenzionalnom grafikom bez uvođenja nove sintakse. Prva verzija stvorena je od 1999. do 2001.

I: Jeste li bili jedina osoba koja je radila na Elici ili je na njezinom razvoju radilo više ljudi?

P: Cijeli projekt Elica uključivao je nekoliko ljudi, ali programiranje sam u cijelosti radio sam.

I: Znate li, je li se Elica ikada koristila za obrazovne svrhe? Ako je, imate li materijala vezano uz to?

P: Da, bila je značajno korištena, ali ne sama Elica, već aplikacije napisane u Elici (koje sam ja napisao). Neki od obrazovnih programa temeljenih na Elici korišteni su na Tehnološkom institutu Stevens; drugi softver temeljen na Elici korišten je u srednjim školama diljem Europe (kao dio međunarodnog europskog projekta); ona (Elica) se također koristi u nekim udžbenicima za 5.-6. razred u Bugarskoj. Osim toga, koristio sam Elicu u nekoliko svojih kolegija na Sveučilištu u Sofiji. Ali više ne.

I: Koji su bili vaši glavni ciljevi tijekom razvoja Elice, od jedne verzije do druge?

P: Opći cilj bio je dodati funkcionalnost koja će omogućiti: (1) brzo prototipiranje obrazovnog softvera; i (2) 3D grafiku za obrazovne svrhe.

I: Kako je Elica primljena?

P: Bila je dobro primljena. Svi su bili oduševljeni onime što Elica može učiniti (ali gotovo nitko nije želio koristiti Elicu).

I: Kakvi su bili vaši daljnji planovi za Elicu, koji se nisu nikada ostvarili?

P: Učiniti je prijenosnom (npr. Za Linux i MacOS, kao i za tablete i pametne telefone); imati online verziju Elica; itd.

I: Zašto je razvoj Elice prekinut?

P: Iz nekoliko razloga: (1) ljudi su bili zainteresirani za korištenje aplikacija temeljenih na Elici, ali vrlo malo njih je bilo zainteresirano za korištenje istinske snage Elice i izradu vlastitih aplikacija; (2) nitko nije bio zainteresiran za potporu daljnjeg razvoja Elice (financijski ili drugim sredstvima); (3) počeo sam imati druge obveze i nisam imao vremena za Elicu; (4) imao sam nekoliko sukoba s Logo zajednicom, i to me natjeralo da prekinem i napustim bilo koji posao koji se odnosi na Logo općenito (programiranje, objavljivanje, prezentiranje i podučavanje).

Imajte na umu - Elica je uvijek bila besplatna, sve su aplikacije Elice uvijek bile besplatne, pa je većina mojih radova na Elici (osim prve dvije godine) bila isključivo volonterska.

I: Zbog čega naziv Elica?

P: Elica je akronim. Izvorno puno ime je Educational Logo Interface for Creative Activities (obrazovno Logo sučelje za kreativne aktivnosti), ali kasnije sam smatrao primjerenijim naziv Educational Logo Interface for Computer Animation (obrazovno Logo sučelje za računalnu animaciju).

I: Što biste nekome tko koristi FMS ili Terrapin Logo preporučili za početak učenja. odnosno korištenja Elice?

P: Imam YouTube kanal s nekoliko stotina animacija (i nekoliko kratkih filmova) napravljenih pomoću Elice. Koliko ja znam, u to vrijeme nijedna druga verzija Loga nije se mogla koristiti za izradu takve računalne grafike. Kako sam izvan Logo zajednice posljednjih 5-6 godina, ne znam što trenutno Logo može učiniti.

I: Imate li svoje ideje za korištenje Elica u osnovnoj ili srednjoj školi?

P: Da. Provedene su u mojim Elica projektima. Jedan od njih bio je u okviru većeg europskog projekta nazvanog DALEST. Radi se o dinamičkoj stereometriji. Elica je korištena za izradu 10 aplikacija. Postoji knjiga o njima.

Transkript intervjua na engleskom jeziku:

I: What prompted you to create Elica in the first place and when was the first version released?

P: I have created several implementations of the Logo language. Elica was one of them. So, it is somewhat a continuation of my previous attempts. The main goal of Elica was to extend Logo with OOP and 3D graphics without introducing new syntax. The first version was created in 1999-2001.

I: Were you the only person that worked on Elica or more people were involved?

P: The whole Elica project involved several people, but the programming was entirely and completely done by me.

I: To your knowledge, was Elica ever used for educational purposes? If so, do you have any related materials?

P: Yes, it was significantly used, however not Elica itself, but applications written in Elica (written by me). Some of the Elica-based educational software was used in Stevens Institute of Technology; other Elica-based software was used in secondary schools across Europe (as a part of an international European project); it is also used in some textbooks for 5-6 graders in Bulgaria. Apart from these, I used Elica in several of my courses in Sofia University. But not anymore.

I: What were your main goals during further development of Elica, from one version to the next?

P: The general goal was to add functionality which will allow: (1) fast prototyping of educational software; and (2) 3D graphics for educational purposes.

I: How was Elica received?

P: It was well received. Everyone was amazed by what Elica can do (but almost no one wanted to use Elica).

I: What were your further plans for Elica, which never came to life?

P: To make it portable (e.g. to Linux and MacOS, as well as to tablets and smartphones); to have online version of Elica; etc.

I: Why was Elica discontinued?

P: Several reasons: (1) people were interested in using Elica-based applications, but very few were interested to use the true power of Elica and make their own applications; (2) no one was interested in supporting further development of Elica (financially or by providing other resources); (3) I started to have other duties and had no time for Elica; (4) I had several conflicts with the Logo community, and this made me to terminate and abandon any work related to Logo in general (programming, publishing, presenting and teaching).

Please, note - Elica was always free, all Elica applications were always free, so most of my work on Elica (except for the first 2 years) was because of my volunteering.

I: Why is Elica called like that?

P: Elica is an acronym. The original full name is Educational Logo Interface for Creative Activities, but sometime later I considered a more adequate name would be Educational Logo Interface for Computer Animation.

I: How would you recommend someone coming from FMS and Terrapin Logo to start using (learning) Elica?

P: I have a YouTube channel with several hundred animations (and a few short films) made with Elica. As far as I know, at that time no other Logo could be used to make such computer graphics. As I'm outside Logo community for 5-6 years now, so I do not know what the current Logos can do.

I: Do you have your own ideas for using Elica in Primary or Secondary Education?

P: Yes. They were implemented in my Elica-based projects. One of them was within the scope of a larger European project called DALEST. It is about dynamic stereometry. Elica was used to build 10 applications. There is a book about them.

3. Crtanje pomoću Loga u hrvatskim školama

U ovom poglavlju ću opisati kako se u hrvatskim školama koje uče djecu programiranju u Logu crtaju geometrijski likovi i poneka tijela. Neću opisivati sve metodičke postupke kojima učenici prvi put nauče crtati nove likove i tijela, no držat ću se logičkog redoslijeda kojim se mogu obrađivati.

Napominjem kako učitelji informatike imaju na izbor hoće li s učenicima raditi programiranje u Logu ili nekom od drugih programskih jezika prikladnih za osnovnu školu.

Kako je FMSLogo najkorištenija verzija programskog jezika Logo u hrvatskim školama, svi kôdovi i crteži u ovom poglavlju su rađeni upravo u FMSLogu.

Prije svega, donosim tablicu osnovnih naredbi (u kraćem obliku) potrebnih za crtanje u Logu:

`fd` *`udaljenost`*	Kornjača se pomiče naprijed za zadanu udaljenost	`pu`	Kornjača ne ostavlja trag
`bk` *`udaljenost`*	Kornjača se pomiče unatrag za zadanu udaljenost	`pd`	Kornjača ostavlja trag
`lt` *`kut`*	Kornjača se okreće ulijevo za zadani kut	`pe`	Kornjača briše trag
`rt` *`kut`*	Kornjača se okreće udesno za zadani kut	`repeat` *`broj`* `[`*`naredbe`*`]`	Ponavlja zadane naredbe *broj* puta
`home`	Vraća kornjaču u početni položaj	`to` *`ime ulazi`*	Započinje proceduru naziva *ime* sa zadanim ulazima
`cs`	Briše ekran i vraća kornjaču u početni položaj	`end`	Označava kraj procedure

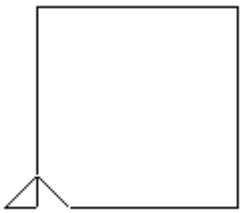
Tablica 3.1: Osnovne naredbe za crtanje kornjačom u Logu

3.1. Crtanje geometrijskih likova

Među prvim geometrijskim likovima koje učenici crtaju u Logu su zasigurno kvadrat i pravokutnik. Prvi crteži još ne koriste naredbu *repeat*, već se učenike usmjerava da likove nacrtaju samostalno pomoću poznatih osnovnih naredbi. Tako bi se, primjerice, kvadrat i pravokutnik mogli nacrtati pomoću sljedećih naredbi:

Kvadrat:

```
fd 100
rt 90
fd 100
rt 90
fd 100
rt 90
fd 100
rt 90
```



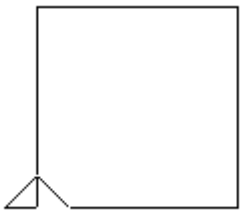
Pravokutnik:

```
fd 100
rt 90
fd 200
rt 90
fd 100
rt 90
fd 200
rt 90
```

Ubrzo nakon toga učenici nauče naredbu *repeat* te njenu korist prilikom crtanja geometrijskih likova. Koristeći naredbu *repeat* isti likovi se mogu nacrtati uz puno manje naredbi:

Kvadrat:

```
repeat 4 [fd 100 rt 90]
```



Pravokutnik:

```
repeat 2 [fd 100 rt 90 fd 200 rt 90]
```

Jednom kada učenici nauče pisati i koristiti procedure, likove možemo crtati i pomoću procedura. Na primjer, procedure za crtanje kvadrata i pravokutnika bi mogle biti napisane na sljedeći način:

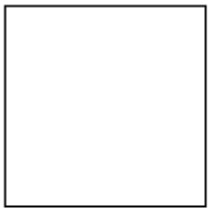
```
to kvadrat :a
  repeat 4 [fd :a rt 90]
end

to pravokutnik :a :b
  repeat 2 [fd :b rt 90 fd :a rt 90]
end
```

Upišemo li nakon definiranja navedenih procedura niz naredbi

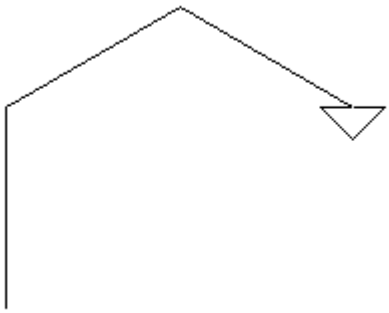
```
kvadrat 100 pu bk 100 pd pravokutnik 150 50
```

dobit ćemo sljedeći crtež:



Učenicima je zatim potrebno zadati da nacrtaju jednakostraničan trokut. Znajući da je veličina unutarnjeg kuta jednakostraničnog trokuta 60°, većina učenika će trokut vjerojatno pokušati nacrtati sljedećom naredbom:

```
repeat 3 [fd 100 rt 60]
```



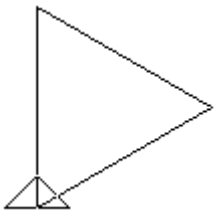
Učenicima je potrebno pojasniti kako se prilikom crtanja geometrijskih likova kornjača okreće za veličinu odgovarajućeg vanjskog kuta te im to zorno pokazati na primjerima u učionici i dostupnim modelima. Zatim ih je potrebno podsjetiti (ili naučiti) kako se računa veličina vanjskog kuta ako je poznata veličina unutarnjeg.

Zatim bi učenici samostalno ili uz minimalnu pomoć trebali moći nacrtati jednakostranični trokut koristeći jednu od sljedećih naredbi:

```
repeat 3 [fd 100 rt 120]
```

ili

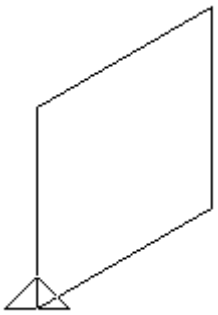
```
repeat 3 [fd 100 rt 180-60]
```



Potom se često crtaju romb i paralelogram, koristeći prethodno naučene naredbe i okretanje kornjače za odgovarajući vanjski kut. Tako bi se, primjerice, romb i paralelogram mogli nacrtati pomoću sljedećih naredbi:

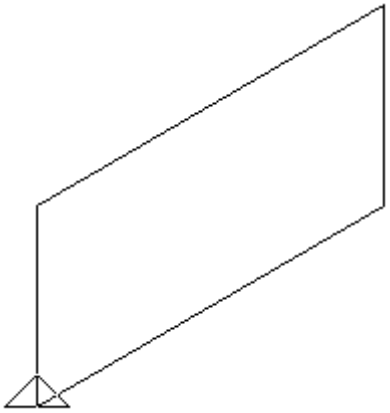
Romb:

```
repeat 2 [fd 100 rt 60 fd 100 rt 120]
```



Paralelogram:

```
repeat 2 [fd 100 rt 60 fd 200 rt 120]
```



Učenicima se također može zadati zadatak da pokušaju nacrtati paralelogram kojem su dvije stranice horizontalne, a ne vertikalne. Primjer takvog rješenja može biti sljedeća naredba:

```
repeat 2 [rt 30 fd 100 rt 60 fd 200 rt 90]
```

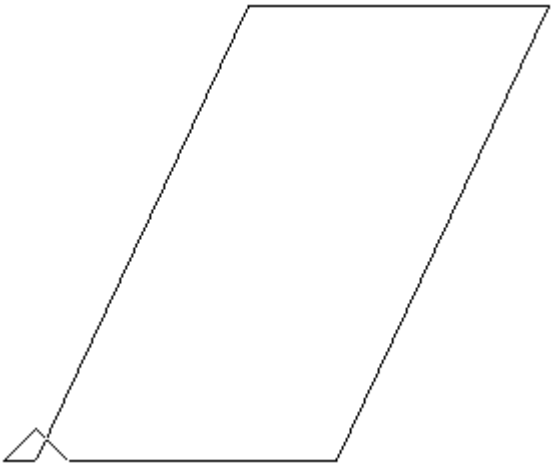
Procedura kojom se može nacrtati paralelogram zadanih duljina stranica i veličine jednog kuta može biti sljedeća:

```
to paralelogram :a :b :kut
  repeat 2 [rt :kut fd :b rt 90-:kut fd :a rt 90]
end
```

Unosom naredbe

```
paralelogram 150 350 25
```

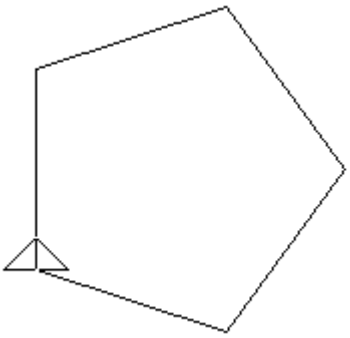
dobiva se sljedeća slika:



Kasnije, nakon što se učenici upoznaju s pravilnim mnogokutima i njihovim svojstvima mogu ih nacrtati i u Logu. Učenici prvo rade crteže s pojedinačnim naredbama za svaki pravilni mnogokut. Slijedi nekoliko primjera.

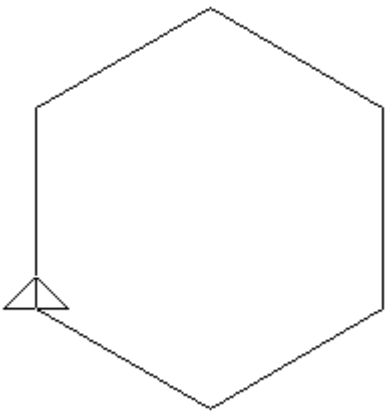
Pravilni peterokut:

```
repeat 5 [fd 100 rt 360/5]
```



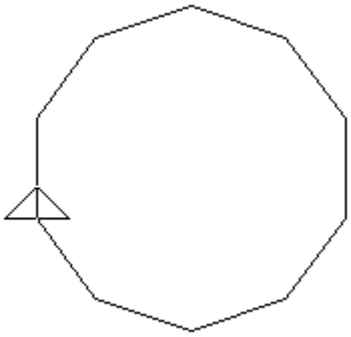
Pravilni šesterokut:

```
repeat 6 [fd 100 rt 360/6]
```



Pravilni deseterokut:

```
repeat 10 [fd 50 rt 360/10]
```



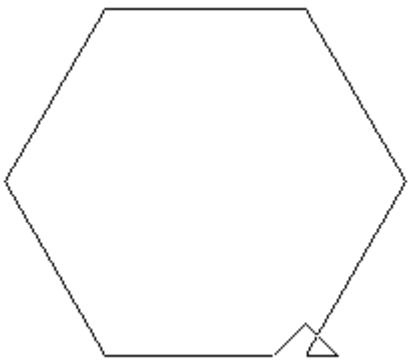
Kako ne bi morali za crtanje svakog novog pravilnog mnogokuta pisati cijelu naredbu, učenike treba uputiti da napišu proceduru kojom će moći nacrtati bilo koji pravilni mnogokut. Primjer takve procedure, uz modifikaciju kojom je donja stranica horizontalno, donosim u nastavku:

```
to mnogokut :n :a
  lt 90
  repeat :n [fd :a rt 360/:n]
  rt 90
end
```

Koristeći navedenu proceduru moguće je nacrtati bilo koji pravilni mnogokut. U nastavku donosim nekoliko primjera:

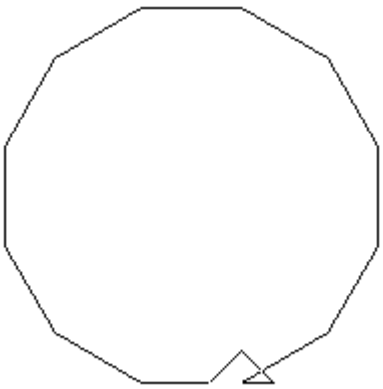
Pravilni šesterokut:

```
mnogokut 6 100
```



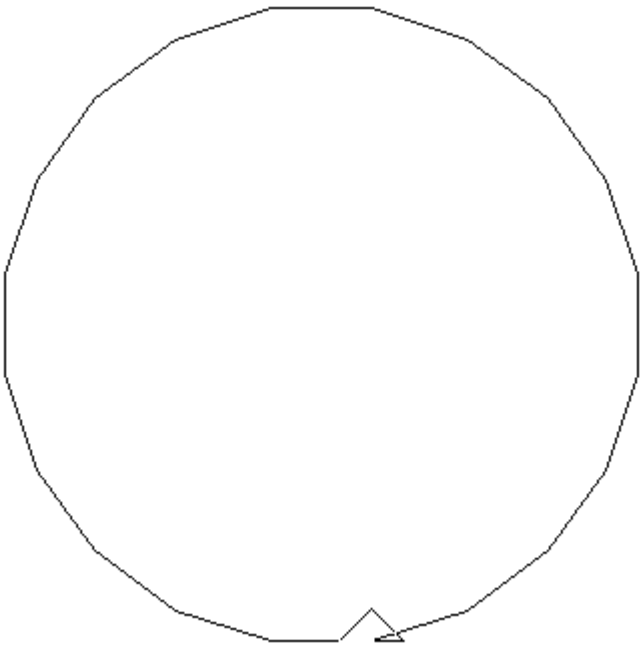
Pravilni dvanaesterokut:

```
mnogokut 12 50
```



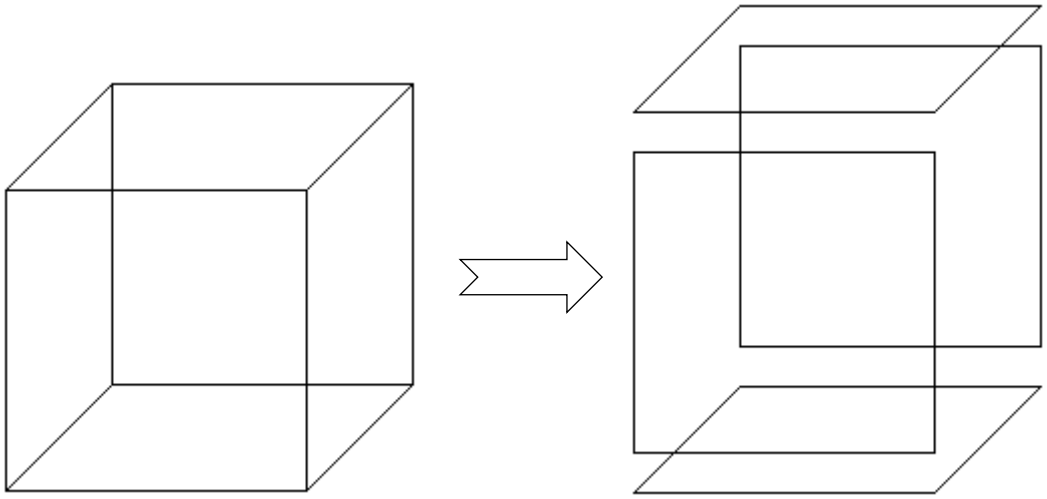
Pravilni dvadeseterokut:

```
mnogokut 20 70
```



3.2. Crtanje geometrijskih tijela

Učenici kocku i kvadar crtaju kao što bi to napravili na papiru, u kosoj projekciji (iako tu terminologiju ne znaju). Bitno im je vizualno predočiti da se crtež kocke zapravo sastoji od dva kvadrata i dva paralelograma:



Učenike je potom potrebno uputiti da samostalno pokušaju nacrtati kocku koristeći postojeće procedure za kvadrat i paralelogram. Nakon što to uvježbaju, trebali bi napraviti novu proceduru kojom je kocku moguće nacrtati samo jednom naredbom.

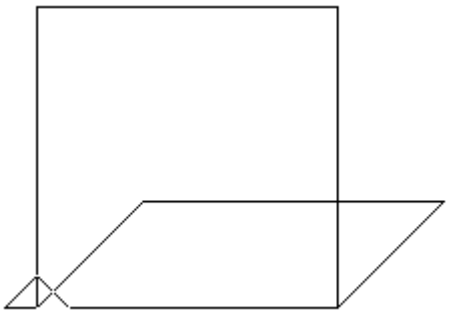
Primjer takve procedure, uz najjednostavniji mogući poziv, donosim u nastavku:

```
to kocka :a
  paralelogram :a :a/2 45
  kvadrat :a
  fd :a
  paralelogram :a :a/2 45
  rt 45 fd :a/2 rt 45
  kvadrat :a
end
```

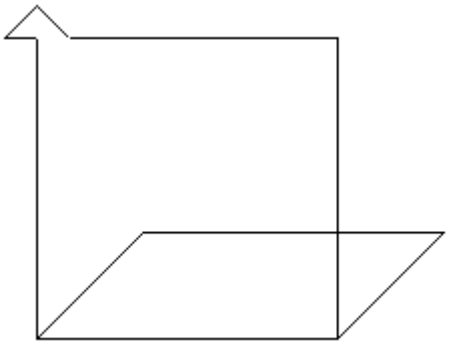
U nastavku ću predočiti izvođenje ove procedure korak po korak:

```
1.) paralelogram :a :a/2 45
```

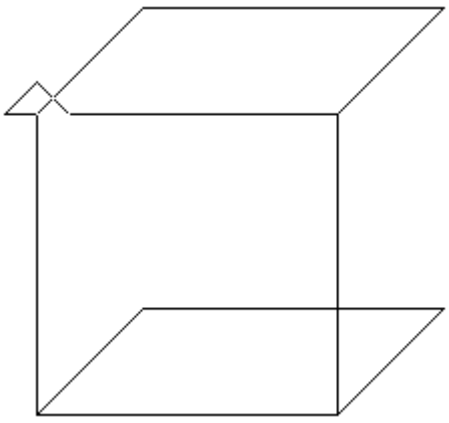
```
2.) kvadrat :a
```



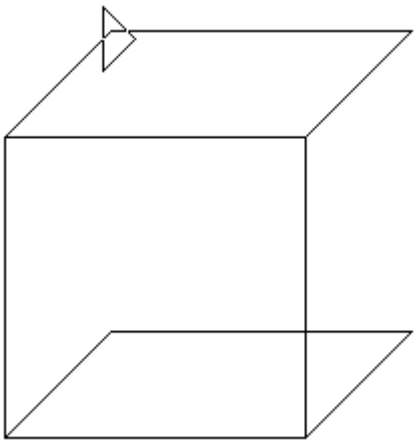
```
3.) fd :a
```



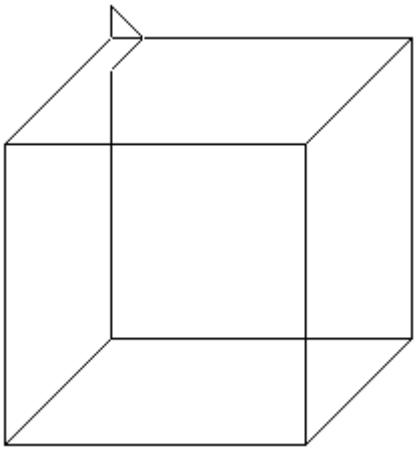
```
4.) paralelogram :a :a/2 45
```



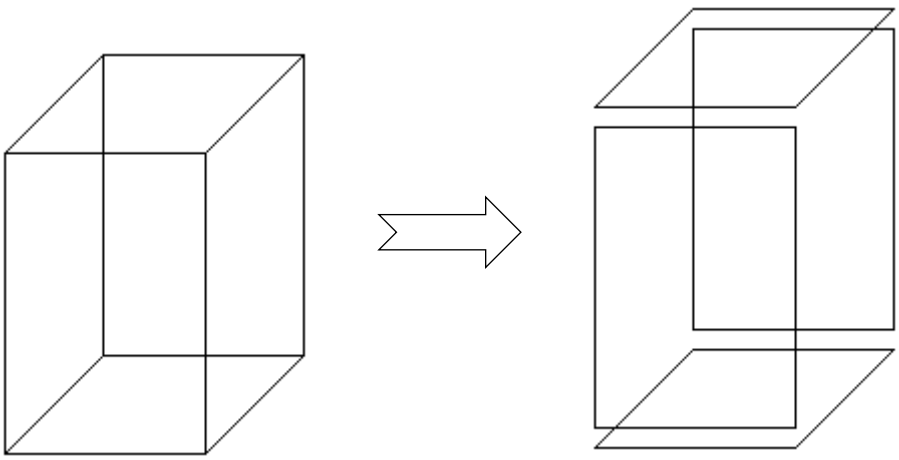
```
5.) rt 45 fd :a/2 rt 45
```



```
6.) kvadrat :a
```



Slično kao kod kocke, učenicima je potrebno zorno predočiti da se crtež kvadra sastoji od dva pravokutnika i dva paralelograma.



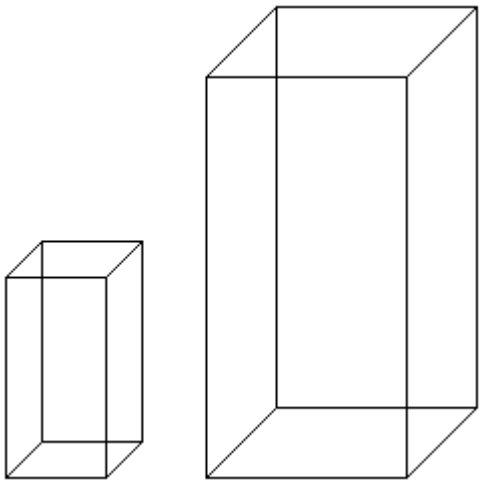
Procedura za crtanje kvadra se može dobiti na način analogan proceduri za crtanje kocke. Duljinu visine kvadra ću fiksirati tako da bude dvostruko dulja od vrijednosti ulaza:

```
to kvadar :a
  paralelogram :a :a/2 45
  pravokutnik :a 2*:a
  fd 2*:a
  paralelogram :a :a/2 45
  rt 45 fd :a/2 rt 45
  pravokutnik 2*:a :a
end
```

Koristeći ovakvu proceduru, naredbe

```
kvadar 50
kvadar 100
```

će nacrtati sljedeće slike:



Vjerujem da je ova procedura dovoljna većini učenika, no naprednijim učenicima je dobro dati zadatak da napišu proceduru s četiri ulaza (duljina, širina, visina i kut) kojom mogu nacrtati kvadar bilo kojeg omjera duljine bridova.

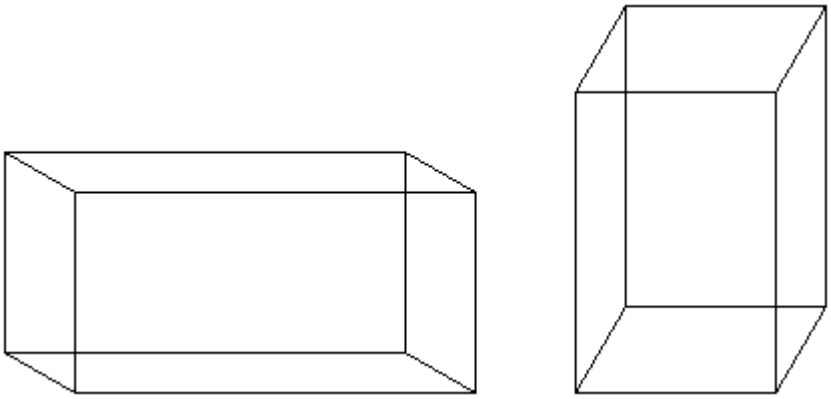
Primjer jedne takve procedure može biti sljedeći:

```
to kvadar :a :b :c :kut
  paralelogram :a :b :kut
  pravokutnik :a :c
  fd :c
  paralelogram :a :b :kut
  rt :kut fd :b rt 90-:kut
  pravokutnik :c :a
end
```

Koristeći novu proceduru za crtanje kvadra, naredbe

```
kvadar 200 40 100 -60
kvadar 100 50 150 30
```

će nacrtati sljedeće slike:



3.3. Trodimenzionalno crtanje u FMSLogu

FMSLogo ima podršku za crtanje u trodimenzionalnom prostoru, koristeći posebne naredbe. Međutim, za ovakav način crtanja potrebno je imati izrazito razvijen prostorni zor, što je rijedak slučaj kod učenika osnovne škole. Slijedom navedenog, crtanje u trodimenzionalnom prostoru se ne obrađuje s učenicima osnovne škole ili se obrađuje samo s najboljim učenicima.

U nastavku je tablica s naredbama za upravljanje kornjačom u trodimenzionalnom prostoru:

`perspective`	Ulaz u trodimenzionalni način rada
`wrap`	Ulaz u dvodimenzionalni način rada
`uppitch` *`kut`* `ili up` *`kut`*	Kornjača uzdiže glavu za zadani kut
`downpitch` *`kut`* `ili down` *`kut`*	Kornjača spušta glavu za zadani kut
`leftroll` *`kut`* `ili lr` *`kut`*	Valjanje kornjače za zadani kut ulijevo
`rightroll` *`kut`* `ili rr` *`kut`*	Valjanje kornjače za zadani kut udesno

Tablica 3.2: Naredbe za upravljanje kornjačom u trodimenzionalnom prostoru

Prikazat ću kako se u trodimenzionalnom prostoru mogu nacrtati kocka i kvadar.

Iako kocka ima 6 strana u obliku kvadrata, za njeno crtanje je dovoljno nacrtati samo 4 kvadrata. Preostale dvije strane će po dovršetku već biti nacrtane stranicama postojećih kvadrata.

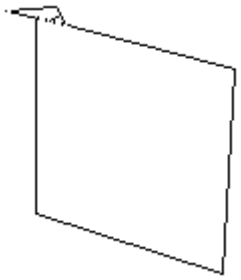
Primjer procedure koja crta kocku u trodimenzionalnom prostoru je sljedeći:

```
to 3D_kocka :a
  perspective
  repeat 4 [kvadrat :a fd :a down 90]
end
```

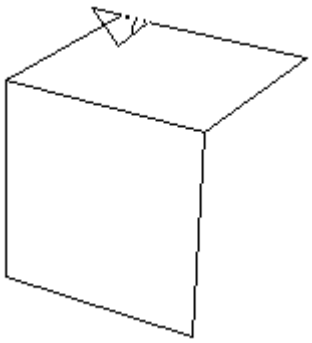
Prikazat ću izvođenje ove procedure, korak po korak, uz poziv naredbe

```
3D_kocka 100
```

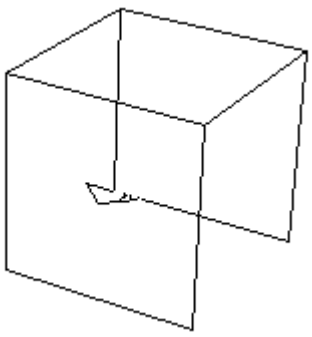
```
1.) Prvi poziv naredbi kvadrat 100 fd 100 down 90
```



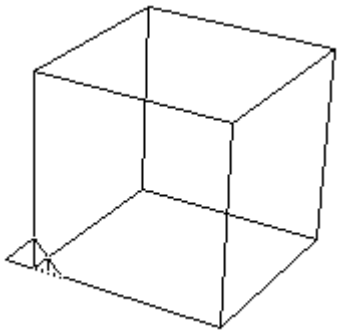
```
2.) Drugi poziv naredbi kvadrat 100 fd 100 down 90
```



```
3.) Treći poziv naredbi kvadrat 100 fd 100 down 90
```



```
4.) Četvrti poziv naredbi kvadrat 100 fd 100 down 90
```



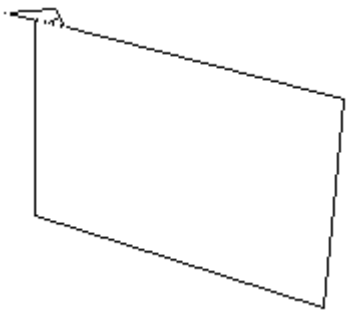
Na sličan način se može nacrtati i kvadar. Primjer jedne takve procedure je sljedeći:

```
to 3D_kvadar :a :b :c
  perspective
  repeat 2 [pravokutnik :a :c fd :c down 90
            pravokutnik :a :b fd :b down 90]
end
```

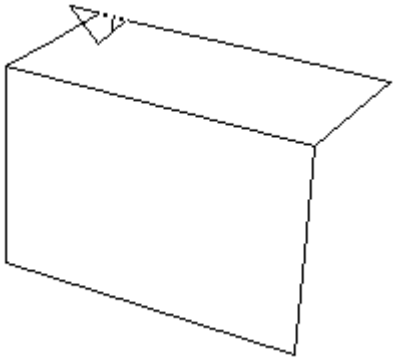
Prikazat ću izvođenje ove procedure, korak po korak, uz poziv naredbe

```
3D_kvadar 150 80 100
```

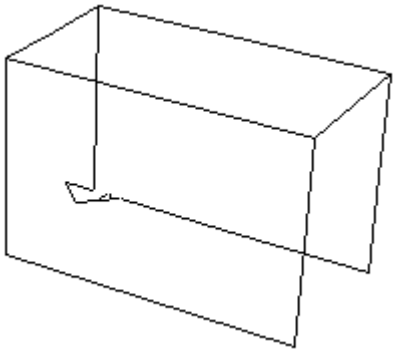
```
1.) pravokutnik 150 100 fd 100 down 90
```



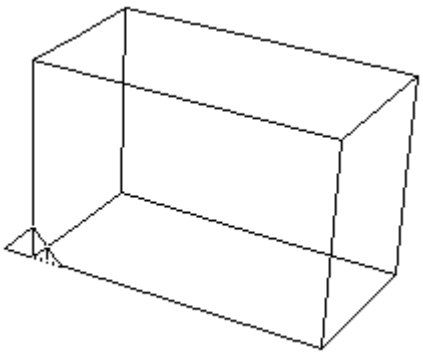
2.) pravokutnik 150 80 fd 80 down 90



3.) pravokutnik 150 100 fd 100 down 90



4.) pravokutnik 150 80 fd 80 down 90



S učenicima je korisno prokomentirati zbog čega je crtanje kocke i kvadra jednostavnije koristeći naredbe za rad u trodimenzionalnom prostoru.

Naime, prilikom rada u trodimenzionalnom prostoru prirodno opisujemo kretanje kornjače dok se sustav brine za prikaz u dvije dimenzije (na ekranu računala). S druge strane, kod izravnog crtanja u dvije dimenzije sami se moramo pobrinuti za prikaz u kosoj projekciji, stoga smo i koristili dodatne naredbe poput

```
rt :kut fd :b rt 90-:kut
```

prilikom crtanja kvadra.

4. Novi načini crtanja u Elici

Iako obustavljena, sve verzije Elice, moderne implementacije programskog jezika Logo, se mogu preuzeti na http://www.elica.net/archive.html. Na žalost, Elica nije bila otvorenog koda te je njen razvoj trajno stao odlukom profesora Boytcheva.

Elica sadrži sve osnovne Logo naredbe te bi i u njoj mogli koristiti procedure opisane u prošlom poglavlju. Međutim, ova implementacija Loga donosi mnoge nove naredbe čije korištenje ću opisati u ovom poglavlju.

Prije svega, donosim tablicu nekoliko novih naredbi za upravljanje kornjačom:

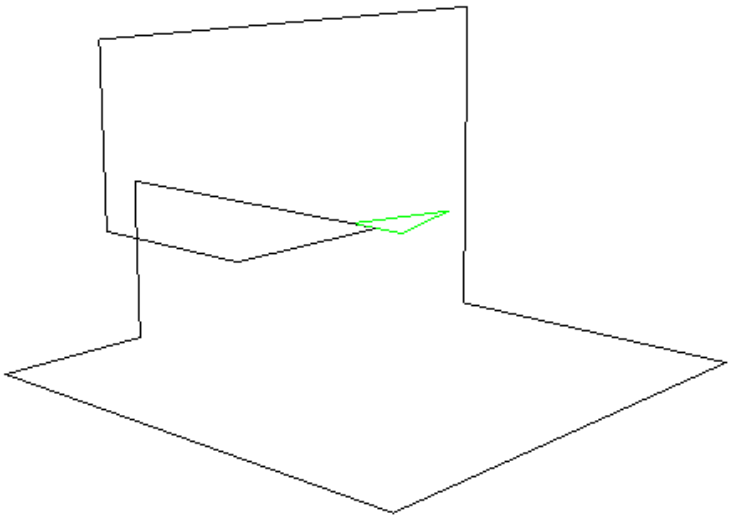
`rise` *`udaljenost`*	Kornjača se penje (pomiče prema gore) za zadanu udaljenost (bez okretanja)
`lower` *`udaljenost`*	Kornjača se spušta (pomiče prema dolje) za zadanu udaljenost (bez okretanja)
`stepleft` *`udaljenost`*	Kornjača se pomiče ulijevo za zadanu udaljenost (bez okretanja)
`stepright` *`udaljenost`*	Kornjača se pomiče udesno za zadanu udaljenost (bez okretanja)

Tablica 4.1: Nove naredbe za upravljanje kornjačom u Elici

U Elici su svi crteži napravljeni u trodimenzionalnom prostoru, bez potrebe za korištenjem naredbe *perspective*. Na primjer, izvršavanjem sljedećeg niza naredbi

```
stepleft 50 lower 25 bk 25 stepright 75 fd 75
stepleft 50 rise 50 bk 75 lower 25 stepright 25 fd 25
```

(i promjenom kuta gledanja naredbom *lookat*, koju ću objasniti kasnije) nastaje sljedeća slika:



Kao što je vidljivo, nove naredbe omogućuju jednostavnije kretanje kornjače prostorom, bez potrebe za promjenom orijentacije kornjače. Procedure za crtanje kvadrata i pravokutnika sada možemo napisati na drugačiji način, koristeći nove naredbe:

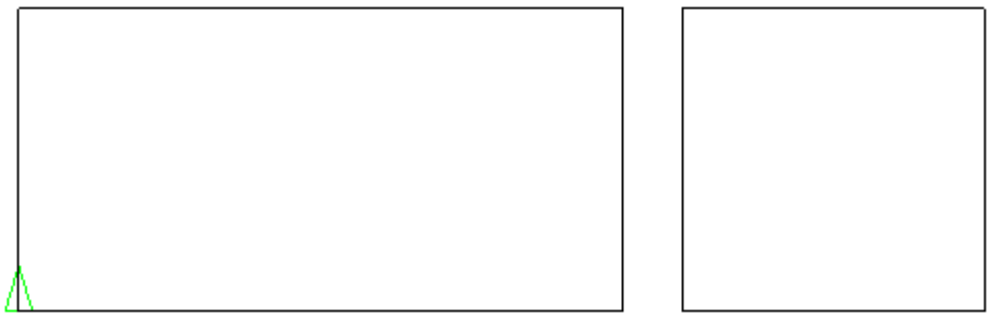
```
to kvadrat :a
  fd :a stepright :a bk :a stepleft :a
end


to pravokutnik :a :b
  fd :b stepright :a bk :b stepleft :a
end
```

Koristeći ovakve procedure, niz naredbi

```
kvadrat 100 pu stepleft 220 pd pravokutnik 200 100
```

će nacrtati sljedeću sliku:



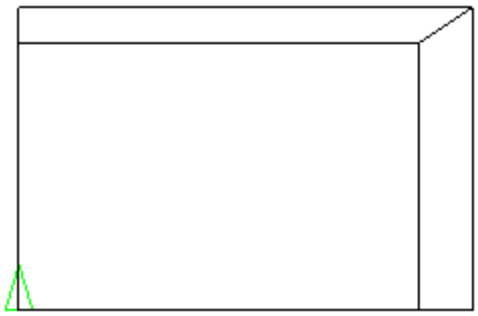
Napravimo sada u Elici proceduru za crtanje kvadra na isti način kao u FMSLogu, uz korištenje naredbe *dn* umjesto *down*:

```
to kvadar :a :b :c
  repeat 2 [pravokutnik :a :c fd :c dn 90
            pravokutnik :a :b fd :b dn 90]
end
```

Unosom naredbe

```
kvadar 150 80 100
```

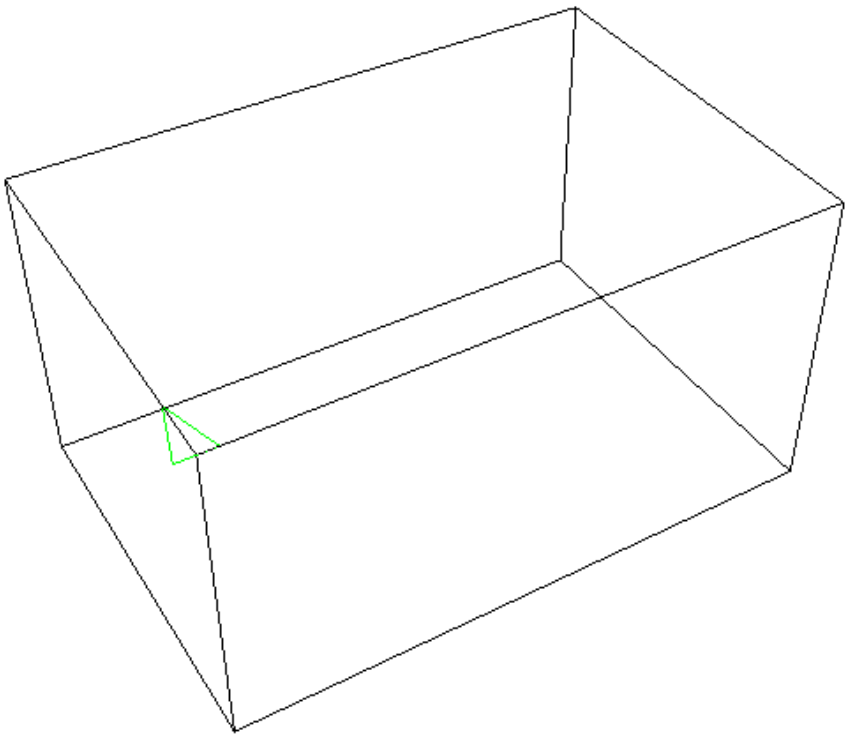
dobiva se sljedeća slika:



Početni kut gledanja nam onemogućuje da vidimo cijeli kvadar u prostoru. Međutim, koristeći novu naredbu *lookat* možemo promijeniti kut gledanja. Primjerice, unosom naredbe

```
lookat vector -50 -20 50 vector 0 -50 0 vector 0 0 1  1  300
```

gdje je *vector -50 -20 50* točka u prostoru s koje gledamo (oko), *vector 0 -50 0* točka u prostoru u koju gledamo (fokus), *vector 0 0 1* vektor koji određuje gdje je „gore“ u prikazu prostora, neobvezna vrijednost varijable *1* koja određuje brzinu promjene kuta gledanja i neobvezna vrijednost varijable *300* koja određuje udaljenost oka i fokusa, dobiva se sljedeća slika:

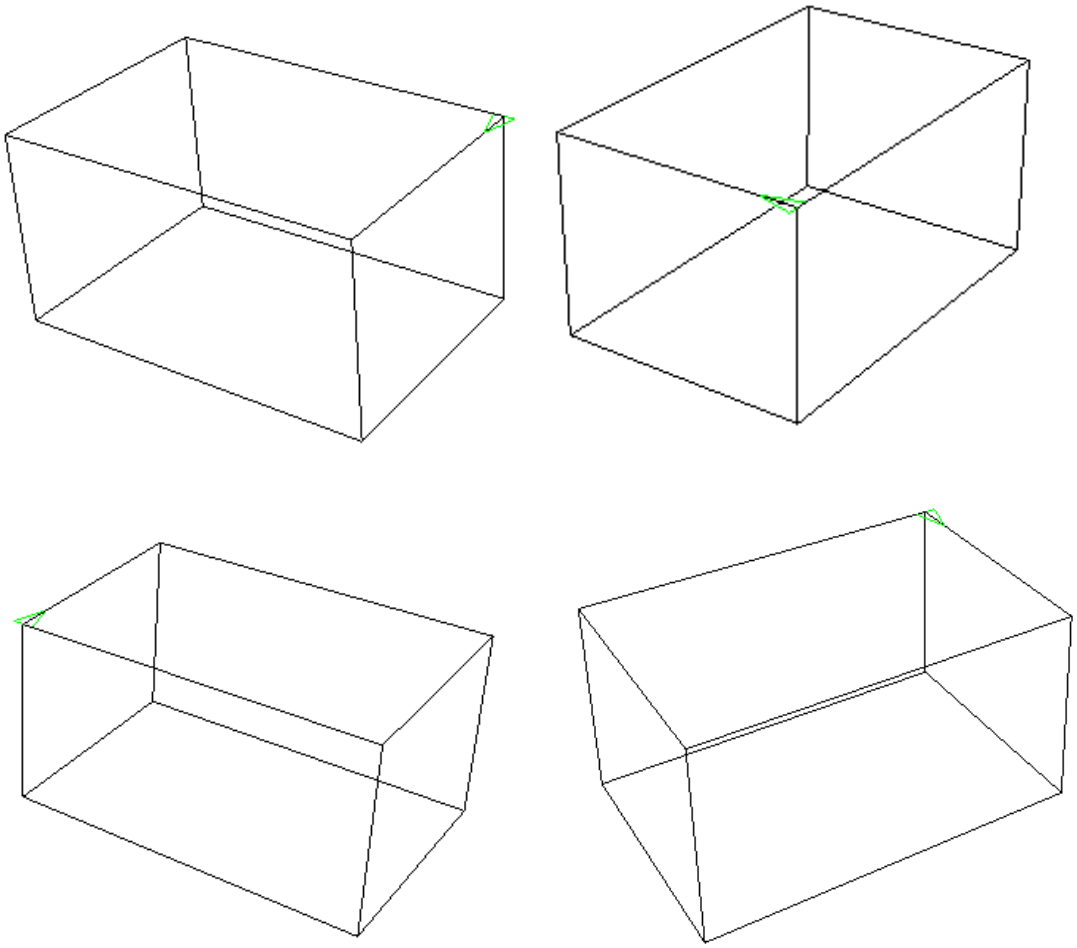


Želimo li napraviti virtualni pregled nacrtanog objekta sa svih strana, možemo iskoristiti novu naredbu *demo udaljenost* koja će kružno pomicati kameru oko središta koordinatnog sustava na zadanoj udaljenosti, time kreirajući animaciju koja nam pruža nove poglede na nacrtani objekt.

Primjerice, zadamo li nakon prethodnog crteža naredbu

```
demo 500
```

dobit ćemo animiranu demonstraciju koja nam pruža, među ostalima, i sljedeće poglede:



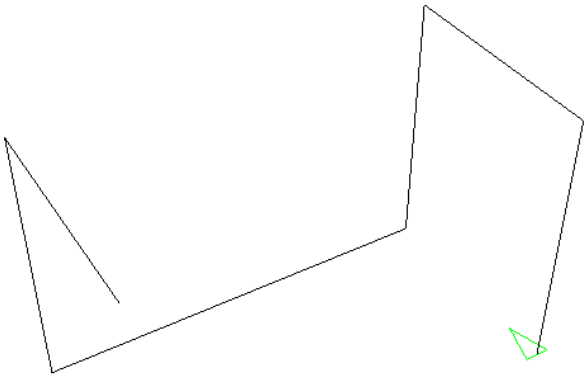
Postojeća procedura za crtanje kvadra je jednostavna i praktična, no prilikom crtanja kornjača čak 8 puta prelazi preko već nacrtanih bridova.

Koristeći nove naredbe dostupne u Elici, sljedećim procedurama također možemo nacrtati kvadar na zanimljive načine, uz samo 3 ponovna prolaska već nacrtanim bridovima:

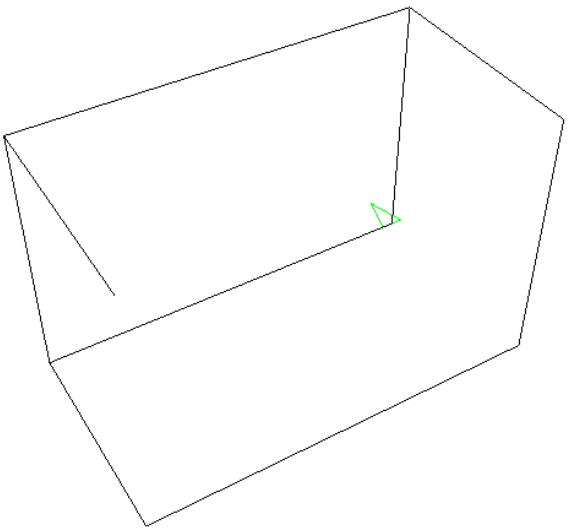
```
to kvadar1 :a :b :c
  fd :b lower :c stepright :a rise :c bk :b lower :c
  stepleft :a fd :b rise :c stepright :a lower :c
  bk :b rise :c stepleft :a lower :c
end
```

U nastavku je prikaz izvođenja navedene procedure naredbom *kvadar1 150 80 100*:

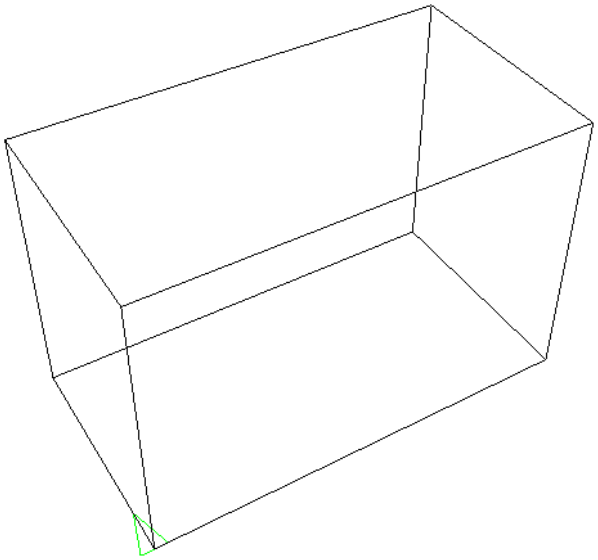
1.) Nakon prvog reda procedure:



2.) Nakon drugog reda procedure:



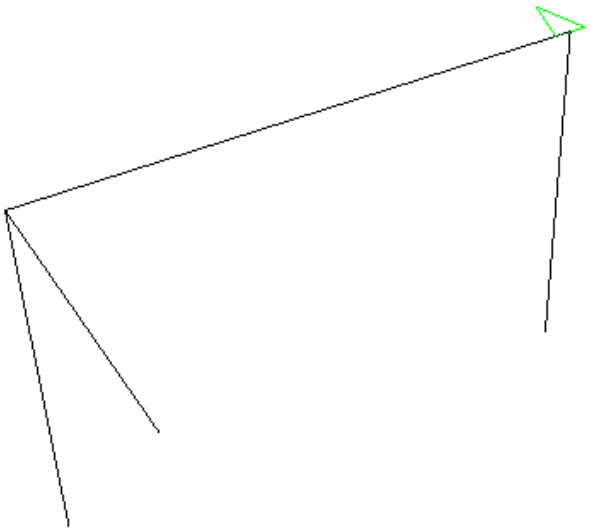
3.) Nakon trećeg reda procedure:



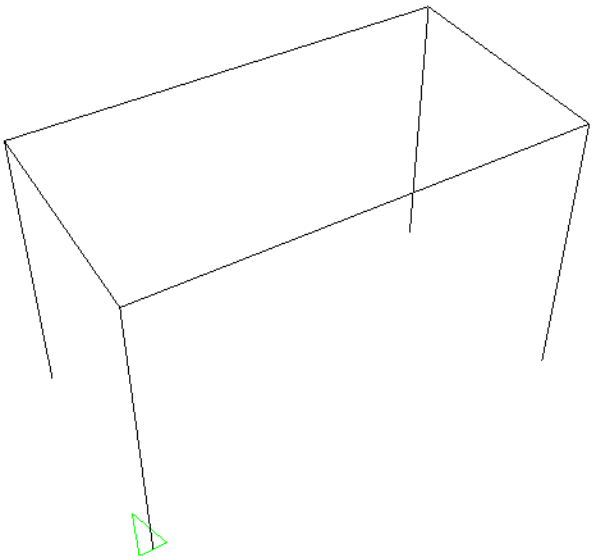
Slijedi nešto drugačije napisana procedura za crtanje kvadra, također s 3 prolaska postojećim bridovima:

```
to kvadar2 :a :b :c
  fd :b lower :c rise :c stepright :a lower :c rise :c
  bk :b lower :c rise :c stepleft :a lower :c
  pravokutnik :a :b
end
```

U nastavku je prikaz izvođenja navedene procedure naredbom *kvadar2 150 80 100*:

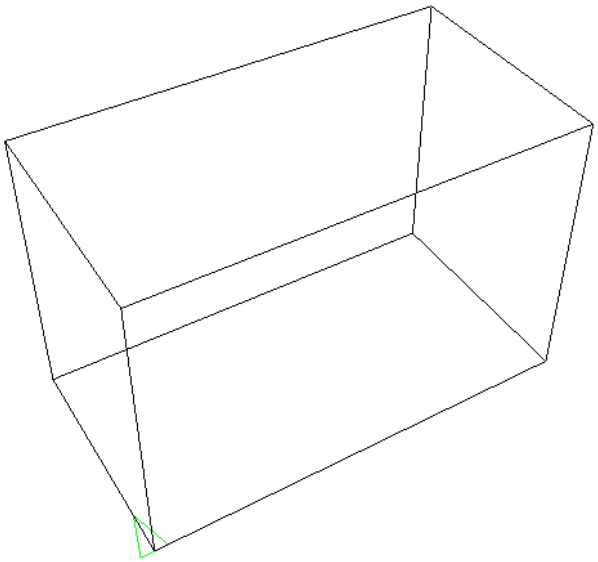


1.) Nakon prvog reda procedure:



2.) Nakon drugog reda procedure:

3.) Nakon trećeg reda procedure:



Samostalna izrada vlastite procedure za crtanje kvadra je odlična vježba učenicima 8. razreda koja povezuje informatiku i matematiku te razvija algoritamski način mišljenja i prostorni zor.

Ako u nastavi programiranja u osnovnoj školi ostane programski jezik Logo, onda bi korisno bilo imati i ovaj jezik proširen naredbama za prirodno crtanje u tri dimenzije. Ovime se učenici uče snalaženju u prostoru, mogu bolje razumjeti pojmove rotacije u tri dimenzije, kao i perspektive. Također, potiče se algoritamsko razmišljanje na način da pri crtanju jednostavnijih objekata u tri dimenzije učenici prepoznaju plohe kojima je objekt omeđen i zadatak crtanja kompleksnijeg tijela rješavaju crtanjem ovih ploha i njihovom pravilnom orijentacijom u prostoru.

Na ovaj se način stereometrijski zadaci mogu bolje razumjeti jer učenici mogu za razna tijela (npr. prizme, piramide) razmišljati na koji su način nastali te prepoznati parametre koji opisuju ovakva tijela.

Stoga smatram da je proširenje jezika Logo naredbama za crtanje u tri dimenzije korisno uvesti već i u osnovnoj školi kako bi djeca, uz crtanje likova u ravnini, na prirodan način i sa sličnim načinom razmišljanja počela crtati i objekte u tri dimenzije.

5. Aplikacije i animacije unutar Elice

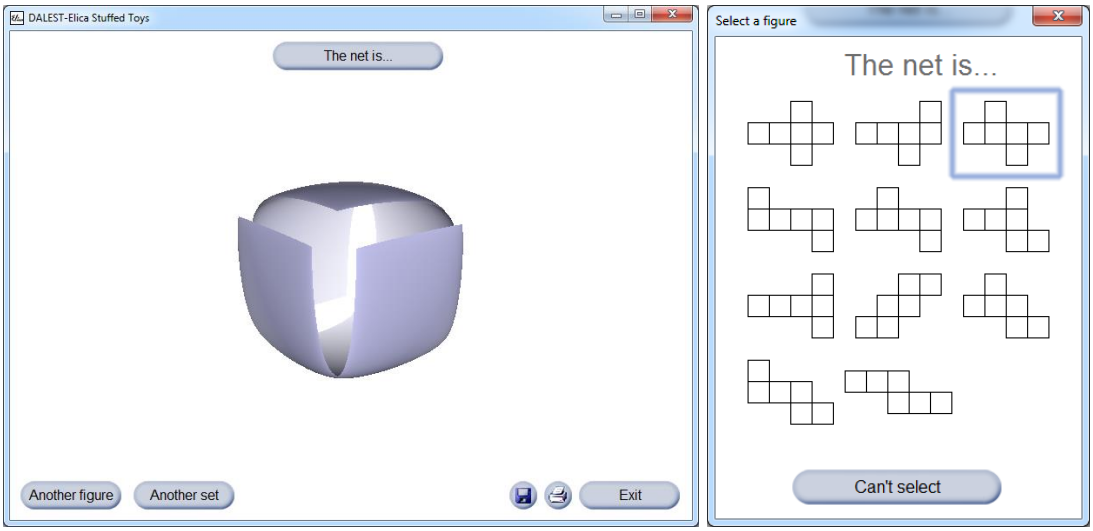
Osim izrade vlastitih crteža i pisanja jednostavnijih programa, Elica pruža mogućnost korištenja gotovih aplikacija i animacija. Naime, unutar Elice se nalazi Muzej (*Museum*) kojem se pristupa kroz izbornik *Help*. Muzej sadrži oko dvadeset interaktivnih aplikacija, uključujući deset aplikacija iz ranije spomenutog projekta DALEST, i stotinjak raznovrsnih animacija. U ovom poglavlju ću opisati neke od uključenih aplikacija i animacija.

5.1. DALEST Stuffed Toys

Kao što joj i ime sugerira, ova aplikacija je rađena za projekt DALEST (Developing Active Learning Environment for Stereometry). Radi se o projektu koji je za cilj imao izradu aplikacija koje bi pomogle učenicima prilikom razvoja prostornog zora i vizualizacije geometrijskih tijela te drugih trodimenzionalnih objekata.

Aplikaciju je, osim unutar Elice, moguće pokrenuti i kao samostalni program u zasebnom prozoru.

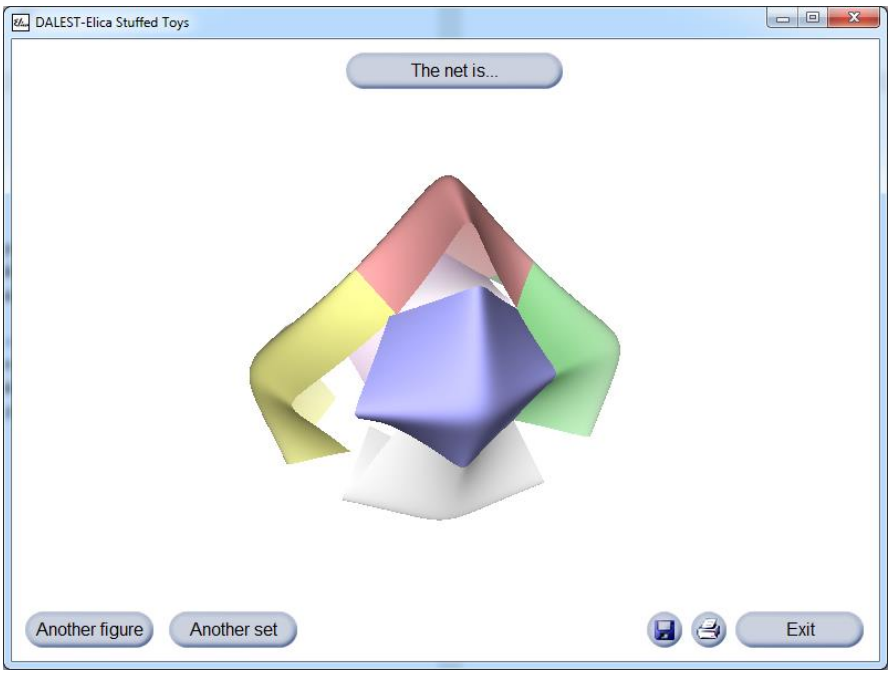
Nakon pokretanja, aplikacija učeniku prikazuje plišanu kocku zaobljenih rubova razrezanu po određenim bridovima. Učenik mišem može okretati kocku kako bi vidio gdje se nalaze rezovi i kako su spojene pojedine strane. Zadatak učenika je odrediti kojoj od ponuđenih mreža odgovara prikazana kocka.



Slika 5.1: DALEST Stuffed Toys – primjer prikazane kocke i ponuđenih mreža

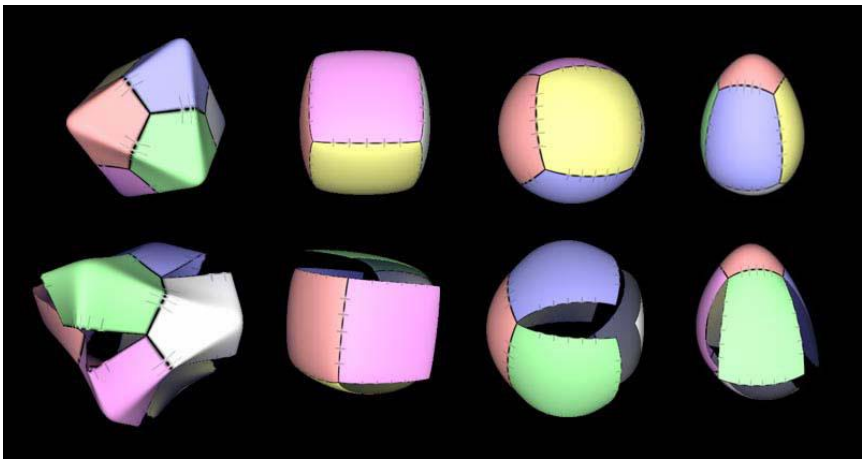
Odabirom ispravne mreže, aplikacija potvrđuje točnost odabira te zadaje novi zadatak. Odabirom pogrešne mreže javlja se poruka „Try again“ (pokušaj ponovno). Klikom na gumb *Another figure* zadaje se novi zadatak.

Klikom na gumb *Another set* zadaju se zadaci po kojima je aplikacija dobila naziv. Prikazuju se razne plišane igračke raznovrsnih oblika, podijeljene na obojane „zakrpe“ od kojih se može složiti mreža kocke.



Slika 5.2: DALEST Stuffed Toys – primjer prikazane plišane igračke

Zadatak učenika ostaje isti, no ovi zadaci su svakako teži zbog raznovrsnosti ponuđenih oblika prikazanih igračaka.

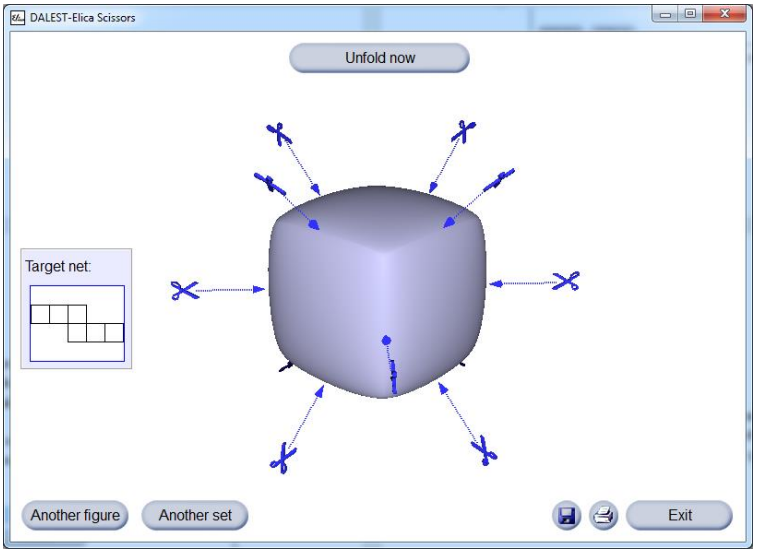


Slika 5.3: DALEST Stuffed Toys – neki od dostupnih oblika plišanih igračaka

5.2. DALEST Scissors

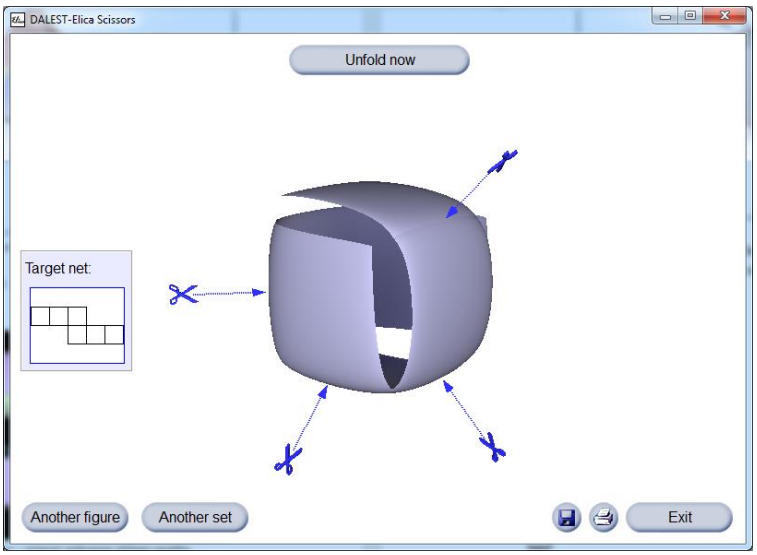
Ova aplikacije je također napravljena za projekt DALEST te je i nju, osim unutar Elice, moguće pokrenuti kao samostalni program u zasebnom prozoru.

Nakon pokretanja aplikacije učeniku je prikazana nerazrezana kocka i zadana mreža.



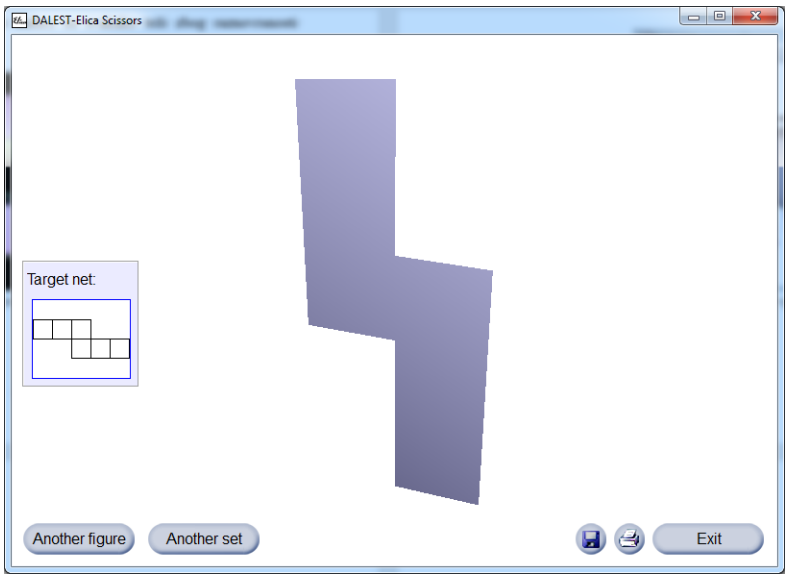
Slika 5.4: DALEST Scissors – primjer zadatka nakon pokretanja aplikacije

Ova aplikacije pred učenike zadaje obratni smjer problema iz prošle aplikacije: razrezati prikazanu kocku kako bi se dobila zadana mreža. Učenici ponovno mišem mogu okretati kocku, a klikom na za to predviđena mjesta kocku razrezuju.



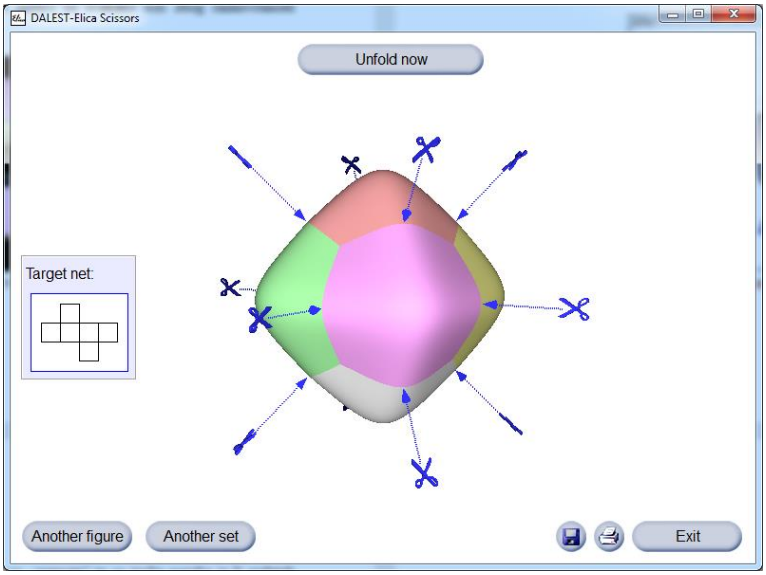
Slika 5.5: DALEST Scissors – primjer razrezivanja prikazane kocke

Klikom na gumb *Unfold now* kocka se „razmota“ te se javlja poruka je li zadatak uspješno ili pogrešno riješen.



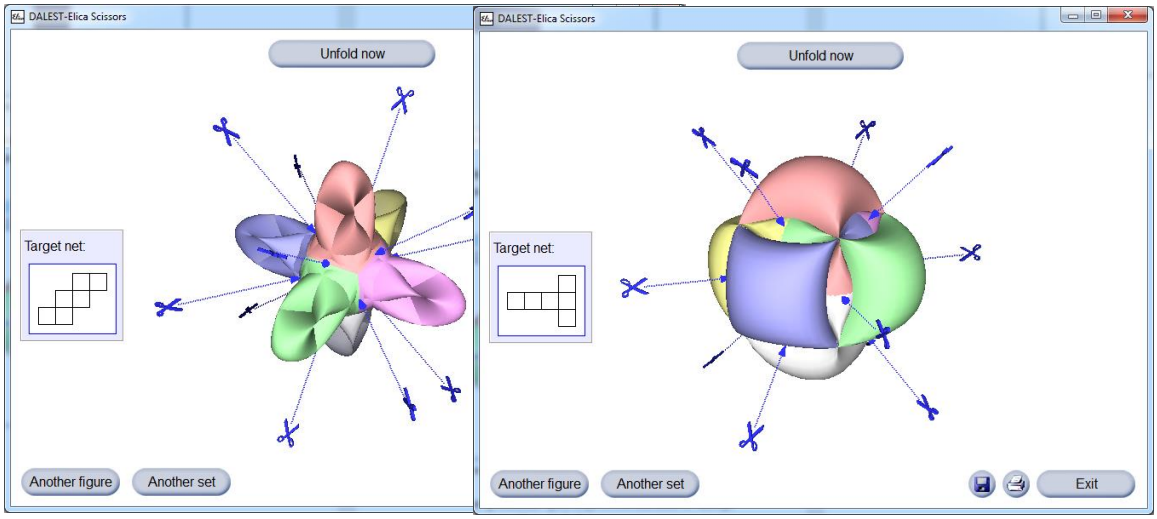
Slika 5.6: DALEST Scissors – „razmatanje“ razrezane kocke

Klikom na gumb *Another figure* zadaje se novi zadatak, a klikom na gumb *Another set* prikazana tijela koja treba razrezati postaju raznolike plišane igračke, već viđene u prethodnoj aplikaciji.



Slika 5.7: DALEST Scissors – nerazrezana plišana igračka

Ponovnim klikom na gumb *Another set* prikazana tijela koja treba razrezati se sastoje od raznobojnih vrpca, što značajno otežava postavljene zadatke.

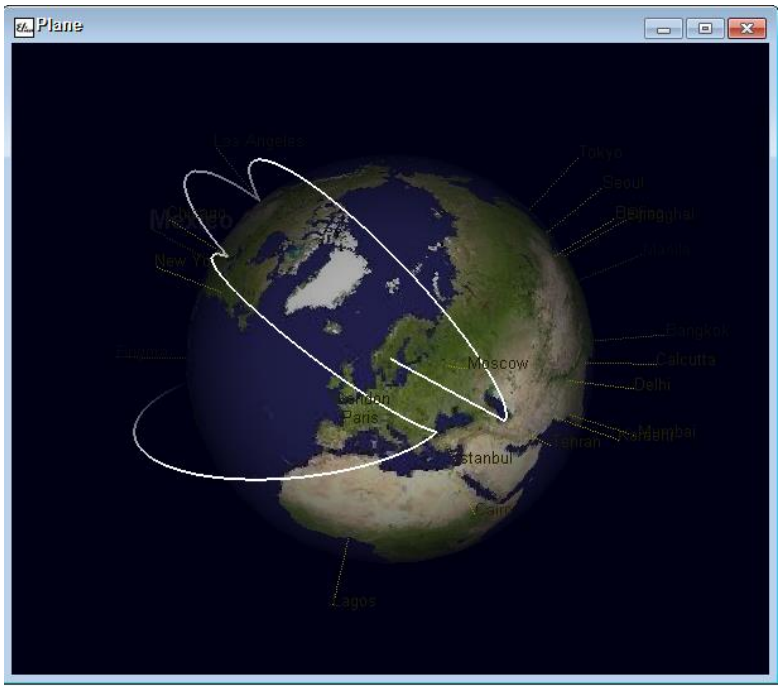


Slika 5.8: DALEST Scissors – tijela sastavljena od vrpci

Obje opisane aplikacije su odličan alat prilikom obrade mreža geometrijskih tijela te na vrlo jednostavan i oku ugodan način učenicima pomažu razviti njihov prostorni zor.

5.3. Around the World

Ova animacija prikazuje simulaciju putanja uzastopnih izravnih avionskih letova između nekih od najvećih svjetskih gradova. Dostupna je unutar Muzeja, u grupi animacija *Scenery*.



Slika 5.9: Around the World – prikaz simulacije putanja avionskih letova

Animacija se može prikazati na nastavi Geografije te pri tome usporediti putanje avionskih letova s poveznicama gradova na karti svijeta. Učenici će uočiti da najkraća udaljenost dvaju gradova na karti ne odgovara putanji najkraćeg leta između tih gradova.

U animaciji je planet Zemlja napravljen pomoću kugle preko koje je stavljena tekstura Zemlje, a redoslijed gradova se odabire nasumice.

U nastavku prilažem kôd ove animacije, kako bih naglasio njenu kompleksnost:

```
; ================================
; "Around The World"
; by Pavel Boytchev (pavel@elica.net)
;
; How to make the trajectory of a plane flying from
; one city to another?
;
; Locations on Earth can be defined by their
; latitude and longitude. However, to draw the
; trajectory of an airplay flying from one position
; to another, it is better to use 3D Euclidean
; coordinates.
;
; If we have the trajectory moving on the Earth
; surface, then taking off and landing can be
; done by adding (vertically) the hump of the sine
; function (scaled appropriately)
;
; This exaple defines some of the biggest cities
; on Earth and simulates a chain of 1000 random
; flights.
; ================================

run "graphix

screen [600x500 [rgb 0 0 20] [fog 150 180]]
make "light "true
make "mode 2
suspendpaint


; Create image of the Earth
make "EarthFontSmall bitmapfont "Arial 16 "false
make "EarthFontBig bitmapfont "Arial 24 "true
make "EarthRadius 30
make "Earth custom sphere vector 0 0 0 :EarthRadius (set
    "color rgb 155 155 155
    "texture texture "Earth.jpg
    "texture.scalex 60
    "texture.scale 30
    "texture.spin 0
    "texture.smooth "false
    "spin 180
```

```
    "smooth "true
)

; Defines an object of type GeoPosition which has latitude,
; longitude and the corresponding Euclidean 3D vector
to City :CityName :Latitude :Longitude
    ; v is position as Euclidean 3D coordinates
    local "v
    to Update
        make "v.x (cos :Longitude)*(cos :Latitude)
        make "v.y (sin :Longitude)*(cos :Latitude)
        make "v.z (sin :Latitude)
    end
    Update

    ; build city's marker
    make local "n custom (text :CityName :EarthFontSmall AboveSurface :v 10) (set "color rgb
155 155 0)
    make local "c custom (segment vector 0 0 0 :n.center) (set
        "color rgb 155 155 0
        "pattern "xoxo)

    ; select a city
    to Select
        make "n custom (text :CityName :EarthFontBig AboveSurface :v 10) (set "color rgb 255
255 255)
        make "c.color rgb 255 255 255
    end

    ; unselect a city
    to Unselect
        make "n custom (text :CityName :EarthFontSmall AboveSurface :v 10) (set "color rgb 155
155 0)
        make "c.color rgb 155 155 0
    end

    to ondrawimage end
end

; Add a new city to the world map
make "Cities [ ] ; list of cities
to AddCity :CityName :Latitude :Longitude
    make :CityName City :CityName :Latitude :Longitude
    make "Cities lput :CityName :Cities
end

; Calculate intermediate point of two geo positions.
; Middle must be from 0 to 1 inclusive.
to GeoVector :Position1 :Position2 :Middle
    ; make v to point to the middle point
    local "v
    make "v.x (1-:Middle)*:Position1.v.x+:Middle*:Position2.v.x
    make "v.y (1-:Middle)*:Position1.v.y+:Middle*:Position2.v.y
    make "v.z (1-:Middle)*:Position1.v.z+:Middle*:Position2.v.z
```

```
    ; normalize v to have length 1
    make local "len sqrt :v.x*:v.x+:v.y*:v.y+:v.z*:v.z ; length
    make "v.x :v.x/:len
    make "v.y :v.y/:len
    make "v.z :v.z/:len
    output :v
end

; Calculate position above Earth surface
to AboveSurface :v :Distance
    make local "k :EarthRadius+:Distance
    output vector :v.x*:k :v.y*:k :v.z*:k
end

; Create some the the biggest cities
AddCity "Tokyo 36 139
AddCity "'Mexico City' 19 -99
AddCity "Mumbai 19 73
AddCity "'Sao Paulo' -24 -47
AddCity "'New York' 41 -74
AddCity "Shanghai 31 121
AddCity "Lagos 6 3
AddCity "'Los Angeles' 34 -118
AddCity "Calcutta 23 88
AddCity "'Buenos Aires' -35 -58
AddCity "Seoul 38 127
AddCity "Beijing 40 116
AddCity "Karachi 25 67
AddCity "Delhi 29 77
AddCity "Manila 15 121
AddCity "Cairo 30 31
AddCity "Jakarta -6 107
AddCity "Paris 49 2
AddCity "Istanbul 41 29
AddCity "Moscow 56 38
AddCity "London 52 0
AddCity "Lima -12 -77
AddCity "Tehran 36 51
AddCity "Bangkok 14 101
AddCity "Chicago 42 -88
AddCity "Bogota 5 -74
AddCity "Canberra -35 149

; Returns the first city then shuffle the rest
to AnyCity
    make local "answer first :Cities
    make local "i 5+round random (count :Cities)-5 0
    make "Cities (se (bf :Cities :i) (first :Cities :i))
    output :answer
end

; Collect traces of flights
make "p 0
make "p1 vector 0 0 0
make "pnts custom set (set
```

```
    "mode 1
    "width 2
    "color (rgb 255 255 255))

; Make a flight from one city to another
to FlyTo :City
    make "FromCity :ToCity
    make "ToCity :City
    ask :ToCity [Select]

    ; handle positions near the poles
    if (abs :(:ToCity)("Latitude))>79
    [
        make (:ToCity)("Longitude) :(:FromCity)("Longitude)
        run (list (:ToCity)("update))
    ]

    ; calculate how many steps to make
    make "x :(:FromCity)("v.x)-:(:ToCity)("v.x)
    make "y :(:FromCity)("v.y)-:(:ToCity)("v.y)
    make "z :(:FromCity)("v.z)-:(:ToCity)("v.z)
    make "d sqrt :x*:x+:y*:y+:z*:z
    make "n round 30*:d+5 0

    ; make the flight
    repeat :n+1
    [
        make "k ((repeat)-1)/:n
        make "gv GeoVector :(:FromCity) :(:ToCity) :k
        lookat :gv vector 0 0 0 vector 0.01 0.01 1 1 175

        make "p0 :p1
        make "p1 AboveSurface :gv 10*(sin 180*:k)
        make "pnts(:p imod 200) segment :p0 :p1
        regenerateimage "pnts(:p imod 200)
        make "p :p+1
    ]

    ask :ToCity [Unselect]
end

; Set initial position
resumepaint
make "ToCity "Bogota
lookat :(:ToCity)("v) vector 0 0 0 vector 0 0 1 1 175

; Make 1000 flights
repeat 1000
[
    FlyTo AnyCity
]
```

5.4. Solar System

Ova animacija, također dostupna u grupi animacija *Scenery*, simulira kretanje planeta u Sunčevom sustavu te se također može prikazati na satu Geografije.

Slika 5.10: Solar System – animacija kretanja planeta u Sunčevom sustavu

Sunce, planeti i Mjesec su napravljeni pomoću kugli preko kojih su stavljene odgovarajuće teksture, a Saturnovi i Uranovi prsteni su zapravo uređeni krnji stošci. Planeti se okreću oko Sunca, a Mjesec oko Zemlje. Također, za svaki planet, Sunce i Mjesec je napravljena animacija rotacije oko njihove osi. Zvijezde su nasumično pozicionirane.

U nastavku prilažem kôd ove animacije, no ovaj put kako bih naglasio korisnost objektnog programiranja:

```
; ================================
;
; "Solar System"
; by Pavel Boytchev (pavel@elica.net) and students
;
; How to simulate the Solar system?
;
; Create each planet (including the Sun and the
; Moon) as a sphere with texture on it. You may
; also consider adding Saturn's and Uranus' rings.
;
```

```
; ================================
run "graphix
screen [800x500 [rgb 0 0 0]]

make "i 0
make "r 50
suspendpaint

make "stars custom set (set "width 1 "color rgb 251 251 0)
repeat 100 [
    make "x (random 2*:r)-:r
    make "maxy sqrt (:r*:r - :x*:x)
    make "y (random 2*:maxy) - :maxy
    make "maxz sqrt (:r*:r - :x*:x - :y*:y)
    make "z (random 2*:maxz) - :maxz
    make "stars(repeat) point :x :y :z
]

make "halo custom (rectangle vector 7-0.1 7.1 14.2  17 17) (set
    "mode 2
    "light "true
    "smooth "true
    "color (rgb 255 155 0 254)
    "texture maskedtexture "SunHalo.jpg "SunHaloMask.jpg
    "focus vector 1 1 0
    "texture.scale 17
)

make "saturnRing custom (cutcone point 0 0 0 5.5 5.5 0.01 0.6) (set
    "mode 2
    "light "true
    "color (rgb 255 255 255 180)
    "hollow "true
    "smooth "true
    "texture maskedtexture "SaturnRing.jpg "SaturnRingMask.jpg
    "texture.scale 1/100
)

make "uranusRing custom (cutcone point 0 0 0 3 3 0.01 0.6) (set
    "mode 2
    "light "true
    "color (rgb 255 255 255 180)
    "hollow "true
    "smooth "true
    "texture texture "UranusRing.jpg "UranusRingMask.jpg
    "texture.scale 1/100
)

to planet :size :surface :rad :astep :sstep
    make local "image custom (sphere point 0 0 0 :size) (set
        "color rgb 255 255 255
        "light "true
        "mode 2
        "smooth "true
        "texture texture :surface
```

```
        "texture.spin 0
        "texture.scale :size
        "texture.scalex 2*:size
    )
    to move
        make "image.spin :sstep*:i
        make "image.center.x :rad*cos :astep*:i
        make "image.center.y :rad*sin :astep*:i
    end
    to ondrawimage end
end

make "sun planet 5 "Sun.jpg 0 0 3
make "mercury planet 0.5 "Mercury.jpg 10.5 3 11.5
make "venus planet 1 "Venus.jpg 13 2.6 -10.5
make "earth planet 1.2 "Earth.jpg 16.2 1.5 10
make "moon planet 0.6 "Moon.jpg 0 0 0
make "mars planet 1 "Mars.jpg 20.6 0.9 8
make "jupiter planet 3.6 "Jupiter.jpg 25.8 0.6 15
make "saturn planet 3 "Saturn.jpg 35.9 0.46 14
make "uranus planet 1.6 "Uranus.jpg 45 0.3 7
make "neptune planet 1.6 "Neptune.jpg 51.2 0.4 7
make "pluto planet 0.4 "Pluto.jpg 56.2 0.2 6

lookat vector 60 60 15 vector 0 0 -10 vector 0 0 1
resumepaint


repeat 3600 [
    suspendpaint

    make "i (repeat)

    sun.move
    mercury.move
    venus.move
    earth.move
    make "moon.image.center.x :earth.image.center.x+2.2*cos 10*:i
    make "moon.image.center.y :earth.image.center.y+2.2*sin 10*:i
    mars.move
    jupiter.move
    saturn.move
    make "saturnRing.center :saturn.image.center
    uranus.move
    make "uranusRing.center :uranus.image.center
    neptune.move
    pluto.move

    resumepaint
]
```

Osim Muzeja dostupnog unutar same Elice, postoji i *Online Elica Museum* na mrežnoj stranici http://www.elica.net/museum/museum.html. Tamo su prezentirane razne aplikacije, animacije i video uradci napravljeni u Elici. Nadalje, na YouTube kanalu ElicaTeam (https://www.youtube.com/user/ElicaTeam) nalazi se oko 250 video uradaka rađenih u Elici, od kojih se mnogi mogu primijeniti u nastavi matematike.

Iako se u Elici mogu napraviti korisne aplikacije i animacije, za to je potrebno puno vremena, volje i znanja. Kako je profesor Boytchev bio jedina osoba koja je radila aplikacije i animacije za Elicu, one se broje (samo) u stotinama. Nezainteresiranost drugih za izradom aplikacija u Elici je u konačnici bio i jedan od razloga prestanka rada na Elici.

Možda u budućnosti postoji šansa da se ovaj projekt oživi jer Elica nudi puno više od samo još jedne u nizu implementacija programskog jezika Logo.

Literatura

A Brief History Of FMSLogo, dostupno na http://fmslogo.sourceforge.net/history (rujan 2018.)

C. Christou, M. Pittalis, N. Mousoulides, D. Pitta, K. Jones, E. Sendova, P. Boytchev, *Developing an Active Learning Environment for the Learning of Stereometry*, dostupno na https://www.researchgate.net/publication/262029642_Developing_an_active_learning_environment_for_the_learning_of_stereometry (rujan 2018.)

P. Boytchev, *Elica Logo and Objects*, dostupno na https://www.researchgate.net/publication/228987995_Elica_Logo_and_Objects (rujan 2018.)

P. Boytchev, T. Chehlarova, E. Sendova, *Enhancing spatial imagination of young students by acticities in 3D Elica applications*, dostupno na https://www.researchgate.net/publication/228375989_Enhancing_spatial_imagination_of_young_students_by_activities_in_3D_Elica_applications (rujan 2018.)

D. Rade, B. Šantalab, L. Novaković, *Like IT 5: udžbenik iz Informatike za 5. razred osnovne škole s CD-om*, Alfa, Zagreb, 2014.

B. Šantalab, K. Toić Dlačić, D. Rade, V. Pilipović, D. Bujadinović, *Like IT 6: udžbenik iz Informatike za 6. razred osnovne škole s CD-om*, Alfa, Zagreb, 2014.

K. Toić Dlačić, D. Rade, L. Novaković, V. Pilipović, I. Matasić, *Like IT 7: udžbenik iz Informatike za 7. razred osnovne škole s CD-om*, Alfa, Zagreb, 2017.

Logo History, dostupno na http://el.media.mit.edu/logo-foundation/what_is_logo/history.html (rujan 2018.)

M. Grinfeld Gradiški, *Logo programiranje: ja hoću i ja mogu programirati na Logu*, vlastita naklada, Zagreb, 1998.

P. Neubauer, *Logo Programming for the Apple II*, dostupno na https://archive.org/details/Kfest2011-PeterNeubaueLogoHistoryAndProgramming (rujan 2018.)

T. Gračanac, *LOGO programski jezik i opis programskog jezika LISP*, Mladost, Zagreb, 1987.

P. Boytchev, *Logo Tree Project*, dostupno na http://www.elica.net/download/papers/LogoTreeProject.pdf (rujan 2018.)

V. Petričević, *Logo za napredne,* vlastita naklada, Vinkovci, 2005.

M. Stančić, B. Vejnović, *Moj Portal 5: udžbenik informatike za 5. razred osnovne škole*, Školska knjiga, Zagreb, 2013.

M. Stančić, B. Vejnović, Z. Dimovski, *Moj Portal 6: udžbenik informatike za 6. razred osnovne škole*, Školska knjiga, Zagreb, 2011.

I. Kniewald, *Programski jezik LOGO*, Multigraf, Zagreb, 1995.

I. Kniewald, *Terrapin Logo*, SysPrint, Zagreb, 2005.

W. Feurzeig, *The LOGO Lineage*, dostupno na http://logoplus.pagesperso-orange.fr/private/The%20LOGO%20lineage.pdf (rujan 2018.)

P. Boytchev, *Using Logo to Model and Animate*, dostupno na https://www.researchgate.net/publication/228868165_Using_Logo_to_model_and_animate (rujan 2018.)

P. Boytchev, T. Chehlarova, E. Sendova, *Virtual Reality vs Real Virtuality in Mathematics Teaching and Learning*, dostupno na https://www.researchgate.net/publication/228373904_Virtual_Reality_vs_Real_Virtuality_in_Mathematics_Teaching_and_Learning (rujan 2018.)

Sažetak

U ovom diplomskom radu sam opisao razloge za stvaranjem programskog jezika Logo i njegov razvoj tijekom prvih nekoliko godina postojanja. Potom sam predstavio dvije implementacije programskog jezika Logo koje se koriste u hrvatskim školama te sam opisao njihov povijesni razvoj. Također sam predstavio Elicu, modernu implementaciju programskog jezika Logo s mnoštvom novih mogućnosti, te proveo intervju s profesorom Pavelom Boytchevom sa Sveučilišta u Sofiji, tvorcem Elice.

U trećem poglavlju sam opisao kako se u FMSLogu, koji se koristi u većini hrvatskih škola, crtaju geometrijski likovi i tijela pomoću kornjačine grafike. Također sam objasnio korištenje naredbi za crtanje u trodimenzionalnom prostoru te predstavio nekoliko procedura za crtanje kocke i kvadra u tom prostoru.

Konačno, opisao sam neke od novih naredbi za upravljanjem kornjačom koje su dostupne u Elici, kao i naredbe za upravljanje kamerom i animirane demonstracije nacrtanih objekata. Također sam predstavio nekoliko aplikacija i animacija uključenih u Elicu koje se mogu koristiti u nastavi Matematike i Geografije.

Summary

In this thesis I described the reasons behind creation of the programming language Logo and provided a brief history of its development during the first few years of its existence. Then I introduced two implementations of Logo used in Croatian schools and I described their development. I also presented Elica, modern implementation of the Logo programming language with a number of new features and conducted an interview with Professor Pavel Boytchev from the University of Sofia, creator of Elica.

In the third chapter, I described how to draw polygons and geometric shapes using turtle graphics in FMSLogo, used in most Croatian schools. I also explained how to use commands for turtle graphics in three-dimensional space and introduced several procedures for drawing cubes and cuboids in that space.

Finally, I have described some of the new commands for control of the turtle available in Elica, as well as commands for camera control and animated demonstrations of drawn objects. I also presented several applications and animations included in Elica that can be used in teaching Mathematics and Geography.

Životopis

Rođen sam u Zagrebu 23.11.1987., a djetinjstvo sam proveo u Samoboru, gdje sam i pohađao Osnovnu školu Bogumila Tonija i Gimnaziju Antuna Gustava Matoša.

2006. godine sam upisao preddiplomski studij matematike, inženjerski smjer, na Matematičkom odsjeku Prirodoslovno-matematičkog fakulteta Sveučilišta u Zagrebu. 2010. godine se prebacujem na nastavnički smjer preddiplomskog studija matematike na istom fakultetu, gdje upoznajem Lanu, buduću suprugu.

2014. upisujem diplomski studij matematike i informatike, nastavnički smjer. Iako sam sve kolegije položio prije ljeta 2016., životni putevi su odlučili odgoditi dovršetak diplomskog rada sve do rujna 2018.

Osim vlastitog školovanja, radio sam u više osnovnih i srednjih škola od rujna 2016. sve do danas. U lipnju 2018. sam imao čast oženiti Lanu, kojoj sam uistinu zahvalan na svom uloženom strpljenju.