Automat za crtanje po školskoj ploči

Tufeković, Anto

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:647553

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-21**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

Automat za crtanje po školskoj ploči

Završni rad

Anto Tufeković

Osijek, 2019.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

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

Osijek, 04.09.2019.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Anto Tufeković
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3999, 25.09.2018.
OIB studenta:	03594098994
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Automat za crtanje po školskoj ploči
Znanstvena grana rada:	**Arhitektura računalnih sustava (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	04.09.2019.
Datum potvrde ocjene Odbora:	11.09.2019.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 12.09.2019.

Ime i prezime studenta:	Anto Tufeković
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3999, 25.09.2018.
Ephorus podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: Automat za crtanje po školskoj ploči**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora

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

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 2

2. PRIMIJENJENE TEHNOLOGIJE 3

2.1. Koračni/steper motor 6

2.2. Sustavi sa koračnim/steper motorima 9

3. REALIZACIJA DVODIMENZIONALNOG V-PLOTERA 11

3.1. Fizički model 11

3.1.1. Problem kod fizičkog modela 13

3.2. Upravljački program 19

3.3. Način funkcioniranja upravljačkog programa 25

3.4. Realizacija i rezultati fizičkog modela 30

4. ZAKLJUČAK 31

LITERATURA 33

SAŽETAK 35

ABSTRACT 36

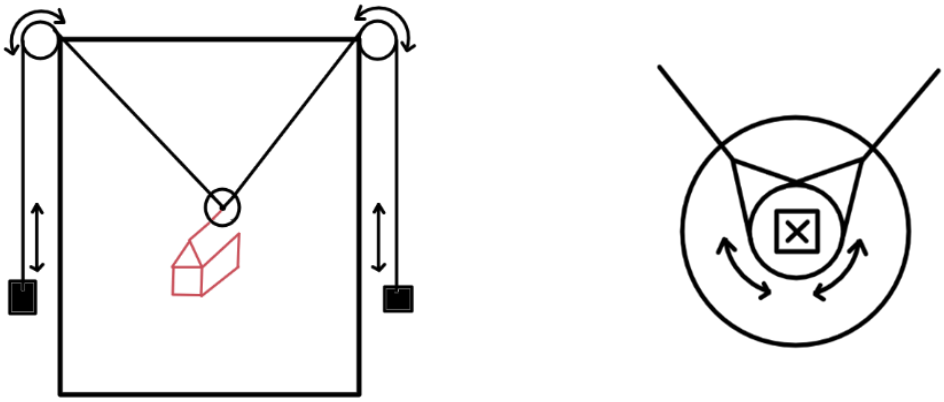
ŽIVOTOPIS 37

PRILOG A: Dodatne slike koje su prevelike za dokument 38

PRILOG B: Python program za izračun napetosti i rezolucije 40

1. UVOD

Cilj ovog završnog rada je automatizirati crtanje po ravnoj površini. To se može postići na više načina: dvodimenzionalno (npr. sa kartezijevim ili polarnim sustavom) ili trodimenzionalno (npr. sa robotskom rukom). Na slici 1.1.(a) je predstavljen pojednostavljeni model polarografa, tj. grafički pisač baziran na biangularnom [1] sustavu (sustav od dva skupa triangulacijska sustava gdje kut veze upravljamo sa dužinom obaju veza). Pisača glava predstavljena na slici 1.1.(b), često zvana gondolom, se spaja na dva motora preko neke veze, kao što su lanci, pojasevi ili užad. Kretnjom motora se te veze skraćuju ili produžuju. Ovakav model polarografa se oslanja na silu težu pisače glave da drži veze napete te mora se postaviti uspravno.



(a) (b)

Slika 1.1. Pojednostavljeni primjer modela (a); Primjer izvedbe pisače glave (b).

Pisača glava mora moći držati crtaći instrument (nešto što ne mijenja dužinu pri korištenju, kao što je npr. marker ili kemijska olovka) u sredini gdje se susreću obje veze. To se ne može fizički napraviti, nego se postavlja da obje veze prividno idu u istu točku. To se izvede tako da se veze spajaju na pisaču glavu preko neke konstrukcije, kao što je standardni kuglični ležaj sa unutrašnjim otvorom dovoljno širokim da može primiti crtaći instrument. Spajanjem na dva međusobno centrirana ležaja omogućuje da se u sredini ležaja postavi crtaći instrument. Time se ne gubi preciznost zbog moguće greške odstupanja. Dodavanje utega pri dnu pisače glave omogućuje da ono uvijek stoji uspravno, što je potrebno ako crtaći instrument odstupa od točne sredine pisače glave. Na drugom kraju veza se dodaju protuteže koje će izjednačiti sile na vezama. Svaki kraj bi trebao biti oko pola težine same pisače glave da se smanji napor na motorima (sustav se oslanja na trenje motora da veze ne kližu) pazeći pritom na konačnu težinu da veze ne popuštaju ili pucaju.

U ovom radu za prototip model se koristi Arduino Uno kao mikroupravljač sustava, te ovisno kako su odabrane komponente (kao što su upravljači za motore) postoji više vrsta sustava. Model koristi

„*Polargraph firmware*“ te se bazira na sustavu kojeg je napravio Sandy Noble iz Škotske [2]. To je sustav koji može primiti više vrsta ulaza, može primiti sliku koju će pretvoriti u sustavu razumljiv oblik ili neku vektor datoteku (npr. *.svg* format) da obradi, prati i crta. Ovaj sustav radi na setu dva koračna motora koji upravljaju dužinom veze. Sa dvije veze imamo dva kuta, te sa promjenom dužine veza mijenjamo nagib oba kuta [3].

1.1. Zadatak završnog rada

U ovom radu je potrebno napraviti uređaj koji crta po školskoj ploči s kredom ili markerom. Potrebno je primijeniti Arduino računalnu platformu, jedan servo motor te dva stepper motora. Uređaj treba ispisivati tekst, odnosno slike, po školskoj ploči. Korisniku treba omogućiti slanje teksta ili slike na izrađeni automat.

2. PRIMIJENJENE TEHNOLOGIJE

Arduino je talijanska tvrtka [4] koja se bavi elektroničkom opremom. Oni proizvode Arduino seriju „kit“ elektronike (Arduino u Americi, Genuino internacionalno) te svu softversku podršku vezanu za nju. Arduino serija ploča su višenamjenski mikroupravljači koji mogu, ovisno kako su programirani, izvesti više svrha, sve od jednostavnog paljenja i gašenja raznih svjetala do upravljanja kompleksnih strojeva kao što su CNC glodalice.

Na Arduinovoj Internet stranici se može preuzeti softverski paket koji služi kao programer Arduino ploča u C jeziku, te preko tog softverskog paketa se kompilira i učitava potreban softver na Arduino pločice.

Arduino Uno, koji je prikazan na slici 2.1., je mikroupravljačka ploča na kojoj se nalazi ATmega328P mikroprocesor koji radi na 16MHz i sadrži 14 ulazno/izlaznih digitalnih pinova, od kojih 6 podržavaju PWM izlaz, te sadrži još dodatnih 6 analognih pinova. Ploča sadrži USB priključak, priključak za napon (7-12V), ICSP pinove i gumb za resetiranje, ostatak specifikacija se nalazi u tablici 2.1.

Tablica 2.1. Specifikacije za Arduino Uno

Mikroupravljač	ATmega328P
Takt procesora	16MHz
Radni napon	5V
Ulazni napon na priključku	7-12V
Istosmjerni napon po ulazno/izlaznom pinu	Do 20mA
Istosmjerni napon po 3.3V pinu	Do 50mA
Količina brze memorije	32KB od kojih 0.5KB zauzima *bootloader*
SRAM	2KB
EEPROM	1KB
Veličina	68.6mm x 53.4mm

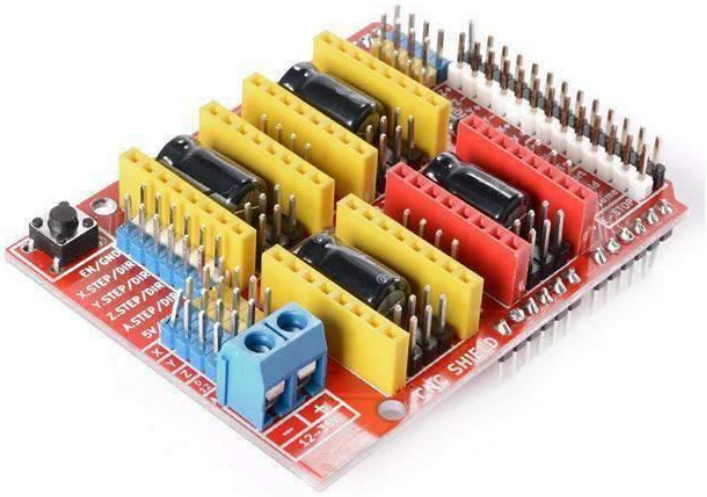
Slika 2.1. Arduino Uno ploča

Arduino serija ploča su dizajnirane na takav način da se lagano mogu izraditi dodatci za njih zvani „štitovi“. Oni su dizajnirani najčešće da se ugnijezde na dostupne pinove.

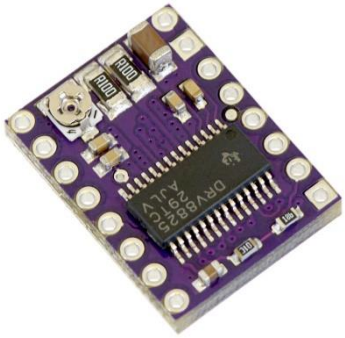
Arduino CNC *shield* (štit) v3.00 je dodatak Arduino Uno pločici [5] čija je glavna uloga da smjesti komponente potrebne za pogon CNC uređaja, a prikazan je na slici 2.2. Sadrži podršku za 3+1 upravljačke komponente koračnih motora, s time da je četvrti motor rijetko korišten ili su mu klonirani koraci od jednog od prva tri motora, te sadrži mogućnost spajanja krajnjih sklopki koje služe za upozorenje korisniku da je stroj otišao van dozvoljenih granica. U ovom projektu se koriste samo koračni motori i jedan servo motor za dizanje i spuštanje crtaćeg instrumenta. Ovaj štit je dizajniran za *Pololu* upravljačke komponente tipa A4988 ili DRV8825. U projektu je korišten DRV8825 koji je prikazan na slici 2.3. i čije su specifikacije navedene u tablici 2.2. [6].

Tablica 2.2. Specifikacije za DRV8825 upravljačku komponentu

Veličina	0.6" x 0.8" ≈ 15.24mm x 20.32mm
Najmanji i najveći radni napon	8.2-45V
Konstantna struja po fazi	1.5A – bez hlađenja
Najveća moguća struja po fazi	2.2A – sa hlađenjem
Najmanji i najveći logički napon	2.5-5.25V
Rezolucija mikrokoračenja	Pun korak (1), 1/2, 1/4, 1/8, 1/16, 1/32 koraka



Slika 2.2. Arduino CNC *shield* (štit) v3.00



Slika 2.3. DRV8825 upravljačka komponenta

Polargraph je set softverskih rješenja koji međusobno komuniciraju te omogućuju upravljanje V-tipa crtaćih uređaja. Sastoji se od software-a koji se nalazi ili na Arduino pločici ili na *Espressif ESP-32* baziranim mikroupravljačima sa *Polarshield* dodatkom koji imaju mogućnost učitavanja datoteka sa SD memorijske kartice. Može primati naredbe direktno od računala preko upravljačkog programa koji šalje naredbe preko USB serijske veze (upravljački program je potreban za rad stroja, ne može Arduino ploča samostalno raditi, dok Espressif ESP-32 bazirane ploče mogu, ali prvo trebaju dobiti potrebnu datoteku od upravljača na SD kartici).

Softver na Arduino pločici sadrži komponente za komunikaciju sa računalom i komponente koje korisnik odredi prije učitavanja samog softvera ovisno o hardveru s kojim korisnik rukuje (npr. kakve upravljačke komponente se koriste za pogon motora). Sadrži još par komponenata koji služe kao pomoćne komponente pri radu stroja.

Python je programski jezik opće namjene i visoke razine kojeg je stvorio Guido Van Rossum [7] u 1991. te značajan je po tome što naglašava čitljivost koda pomoću uvlačenja redaka i korištenjem

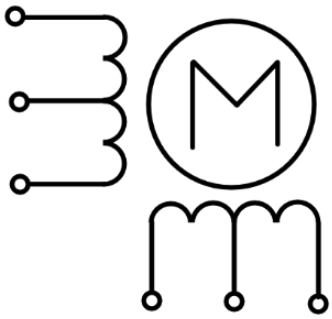
praznih mjesta u redcima, tj. ne koristi vitičaste zagrade nego povećanje uvlačenja označava početak novog ugniježđenog bloka, dok smanjenje uvlačenja označava kraj tog bloka. Koristi princip „pačjeg pisanja“ gdje interpreter automatski zadaje tip podatka zasebnim varijablama ovisno kako se koriste (naprimjer ako se varijabli doda cijeli broj, tip će biti cjelobrojni broj *int*, ako se doda niz slova onda će poprimiti tip *String*, u konačnici korisnik ne zadaje tipove podataka).

2.1. Koračni/steper motor

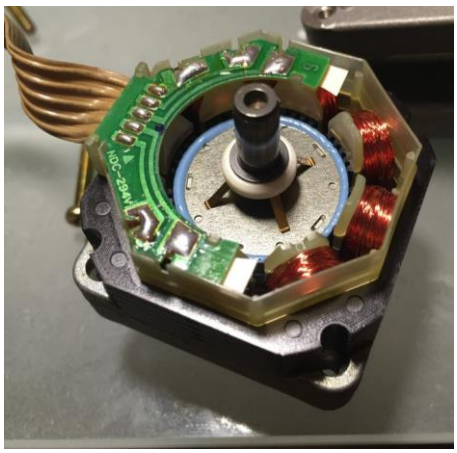
Koračni motori su tip istosmjernih motora bez četkica koji rastavljaju svoje okretanje u određen broj koraka (najčešće se mjeri u stupnjevima ili koracima, npr. 1.8° → 200 koraka za puni okretaj). Njihova osovina se okreće tako da uvijek prati te korake (sa jednim korakom će se uvijek okrenuti za nominalni broj stupnjeva) i to čini bez praćenja aktualne pozicije, nema senzora ili ikakve povratne informacije, radi na sustavu otvorene petlje. Ovi motori mogu tako djelovati dokle god se ne prijeđe preko graničnih karakteristika motora (najvažniji su maksimalna brzina i moment sile koje motor iziskuje).

Postoje dvije izvedbe zavojnica koračnih motora: unipolarno i bipolarno:

Unipolarni motori imaju jedan navoj sa središnjim spojem za svaku fazu, što je prikazano na slici 2.4. Ovom izvedbom možemo svaku fazu pogoniti u oba smjera, te najčešće se središnje spojeve uzemljava da bi se mogao postići ovaj efekt. Ova izvedba je lakša za korištenje jer je jednostavnije za pogoniti koristeći mikroupravljač, ali daje manje snage nego bipolarni jer u svakom danom trenutku koristi samo pola svojih navoja. Često dolaze sa izvedbom od 5 žica, gdje su oba središnja spoja spojena i izvedena kao jedna žica zbog jednostavnosti.



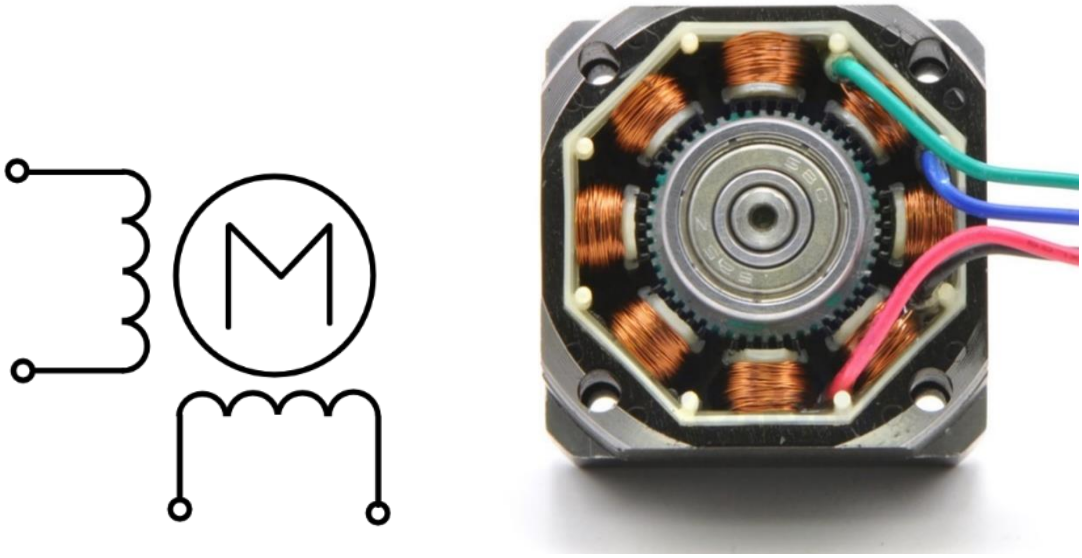
(a)



(b)

Slika 2.4. Model unipolarnog koračnog motora (a); Unutrašnji prikaz unipolarnog motora (b).

Bipolarni motori imaju jedan navoj za svaku fazu što je prikazano na slici 2.5. Ovom izvedbom se mora struja kroz cijeli navoj obrnuti da bi postigli isti učinak kao kod unipolarnih. Istovremeno, ovo pojednostavljuje konstrukciju motora jer ne zahtjeva navoje sa središnjim spojem. Postoje varijacije izvedbi za bipolarne motore: više parnih faza koje su spojene serijski (zahtjeva manju struju, ali imaju veću induktivnost) ili paralelno (zahtjeva više struje, ali imaju manju induktivnost) koje određuju induktivnost i ponašanje pri višim frekvencijama i mogu imati jedan zaseban navoj po fazi ali su onda skuplji za izvedbu i zahtijevaju posebne pogonske strujne krugove da bi mogli raditi. Najčešće dolaze kao motori sa 4 žice, gdje svaki par žica označava jednu fazu sa nekim brojem navoja u nekoj izvedbi.

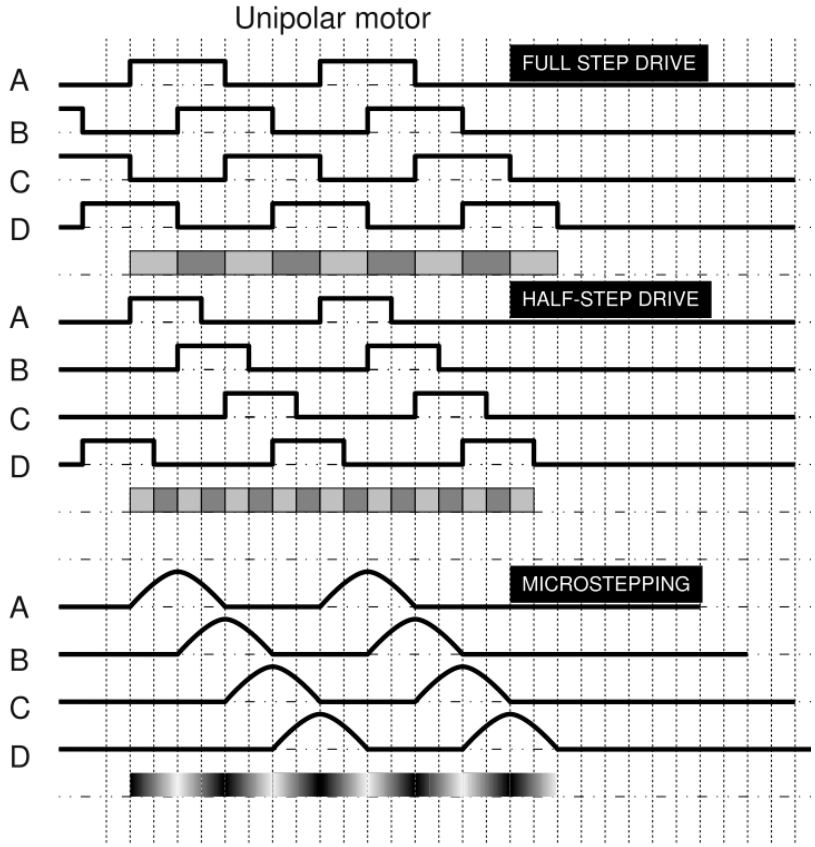


(a) (b)

Slika 2.5. Model bipolarnog koračnog motora (a);
Unutrašnji prikaz bipolarnog motora (b)

Mikrokoračenje (engl. Microstepping) koračnih motora je način pogona koračnih motora na takav način da omogući veću rezoluciju koraka nego što motor već posjeduje (npr. koristeći polukorake od 1.8°/200 koraka dobije se 0.9°/400 koraka). Ovakav način rada se dobije npr. modulacijom signala koji odlazi u koračne motore, najjednostavnije je da se dva sinusna signala pomaknuta fazno za 90° šalju u svaki navoj redom, time se oštra kretnja od jednog koraka pretvori u glatku kretnju. Ovakav učinak nije moguć u stvarnosti jer upravljači za koračne motore ne mogu stvarati savršene sinusne signale, neko aproksimirane, što uzrokuje trzanje, iako puno manje primjetno nego bez mikrokoračenja. Na ovaj način se može samo ugladiti kretnja, ako se želi dobiti više koraka (dva, četiri, osam, …, puta više koraka) mora se na drugačiji način slati signal u koračne

motore, npr. ako se želi dobiti dva puta više koraka mora se preko *PWM* (eng. *Pulse Width Modulation*, modulacija širine impulsa) modulacije specifično mijenjati napon na navojima. To se postiže tako da se u prvom koraku prvom navoju da cijeli napon, u sljedećem koraku *PWM* modulacijom se da pola napona prvom i drugom navoju te na kraju se da drugom navoju puni napon. Takvom manipulacijom rotora se napravi dva koraka umjesto jednog. Ovaj učinak se može dijeliti u više koraka tako da se napon još više dijeli i time postižemo više mogućih koraka kao što je prikazano na slici 2.6.



Slika 2.6. Grafički prikaz napona pri raznim stupnjevima mikrokoračenja [8]

Jedan problem mikrokoračenja je to što svaki korak ima manji moment sile nego kad je bez mikrokoračenja[9]:

$$T_{INK} = T_{PK} \cdot \sin\left(\frac{90}{SMK}\right) \ [Nm] \tag{2-1}$$

Gdje je:

- T_{INK} → inkrementalni moment sile koji svaki mikrokorak proizvodi (Nm)
- T_{PK} → moment sile punog koraka (Nm)
- SMK → stupanj mikrokoračenja, tj. broj mikrokoraka po jednom koraku (2, 4, 8, 16, 32, …)

Inkrementalni moment sile koji svaki mikrokorak proizvodi u formuli 2-1 je važan podatak jer ako je prenizak može se dogoditi da sustav ne daje dovoljan moment sile da nadvlada statičko trenje mehanizma. U tablici 2.2. su izlistane proporcije za stupnjeve mikrokoračenja.

Tablica 2.2. Proporcije momenta sile i mikrokoračenja

Korak mikrokoračenja	1	2	4	8	16	32
Konačni postotak momenta sile	100%	70%	38%	20%	10%	5%

Tablica 2.3. Specifikacije NEMA17 42SHD0001 koračnog motora

Model	42SHD0001
Tip koračnog motora	Bipolarni hibridni
Radni napon	5-24V
Zakretni moment držanja	3.2kg.cm ≈ 0.312Nm
Struja po fazi	1.33A
Otpor faze ±10%	2.1Ω
Kut koraka	1.8°
Induktivnost faze (1Khz)±20%	4.5mH

U prototip sustavu koji koristi koračne motore sa specifikacijama zadanim u tablici 2.3. sa mikrokoračenjem od 32 dalo bi $\rightarrow T_{INK} = 0.312 \cdot \sin\left(\frac{90}{32}\right) = 0.0153\ [Nm]$ što je oko 5% originalnog momenta sile, ali je i dalje dovoljno jako da glatko pogoni sustav zbog protuteža.

2.2. Sustavi sa koračnim/steper motorima

Sustavi upotrebljavaju koračne motore da bi postigli visoku preciznost pokreta a da se pri tome ne mora brinuti dali je pravilno izvršen pomak, jer ako je sustav pravilno dizajniran ono će koristiti koračni motor pravilnih specifikacija (broj koraka po okretaju, maksimalna snaga itd.) te onda ne mora pratiti poziciju motora usred rada (za razliku od konvencionalnih motora gdje ako se želi pratiti pomak mora se koristiti neka povratna veza npr. preko sustava optičkog enkodera).

Prednosti koračnih motora:

- svestrani su jer se mogu u razne svrhe koristiti, od upravljanja robotskih ruka u tvornicama do upravljanja čitače glave u tvrdom disku ili čitaču kompaktnih diskova
- koračni motor istih dimenzija kao konvencionalan motor ima veći moment sile pri nižim brzinama
- nemaju nikakve trošne dijelove osim ležajeva što bi i konvencionalni motori imali te su pouzdani zbog toga
- sadrže grešku jednog koraka do 5%, ali nije ponovljivo kroz više koraka te time postižu visoku preciznost i ponovljivost koraka
- kad se motor pod naponom ne kreće ima punu snagu motora što zaključava kretnju
- jednostavnost motora daje jeftiniju proizvodnju i mogućnost rada u sustavu sa otvorenom petljom

Nedostatci koračnih motora:

- znaju biti glasni u operaciji
- što se brže mora kretati, to manji moment sile iziskuje te može gubiti korake na taj način
- naspram konvencionalnih motora koji koriste sustav zatvorene petlje nema upozorenja ako izgube korak usred operacije
- ovisno o namjeni su neprecizni bez mikrokoračenja
- pri specifičnim frekvencijama upravljačkog sklopa motori mogu rezonirati i titrati oko jednog pola, efektivno stanu sa kretnjom onda
    - ovo se najčešće događa motorima bez opterećenja

3. REALIZACIJA DVODIMENZIONALNOG V-PLOTERA

Rad se sastoji od dva dijela: fizičkog sustava i softverskog upravljača.

3.1. Fizički model

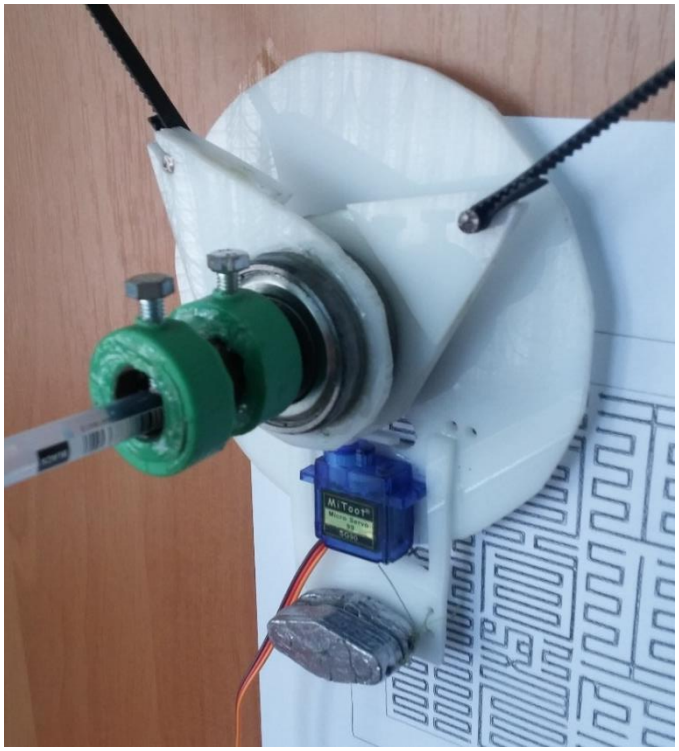
Fizički model se sastoji od:

- ploče na kojoj je sve montirano, ploča mora biti dovoljno čvrsta da se ne savija dok sustav radi, mora imati poželjna površinska svojstva ovisno o vrsti crtaćeg instrumenta (naprimjer ako se crta direktno na ploči, poželjno je da je površina od glatke plastike za pisanje markerom za bijele ploče, ako se neki materijal kao papir lijepi na ploču, poželjnije je da je od nekog grubljeg materijala kao što je drvo da se privremeno ljepilo bolje primi), te mora biti dovoljno velika da bi imali što više efektivnog crtaćeg prostora
- dva koračna motora, ne moraju nužno biti isti ali je poželjno da su sličnih karakteristika, s dovoljnim momentom sile da bi pogonile sustav
- veze kojim spajamo motor i pisaču glavu, moraju biti dovoljno dugačke te biti što manje rastezljive, da se ne bi pojavile distorzije u crtežu
- pisače glave koja se spaja na veze te visi na ploči, mora biti napravljena tako da drži crtaći instrument što je bolje okomito (tako da je dovoljno široka i visoka) bez da bude preteška
- Arduino Uno mikroupravljača koji upravlja sve aspekte fizičkog modela
- računala s kojim se šalju naredbe Arduinu preko serijske veze

Model funkcionira koristeći protuteže, na drugom kraju svake veze se nalazi uteg koji je oko pola težine pisače glave. Oni smanjuju napor na motorima te pružaju gladak pokret po ploči dok sustav crta. Bez njih bi se motori naprezali sa punom težinom pisače glave, uvelo bi trzanje pri crtanju jer je težina samo u jednom smjeru okretanja motora, te zakompliciralo bi konstrukciju jer se mora uvesti neki mehanizam koji ne dopušta klizanje veza pod težinom pisače glave.

Pisača glava (često zvana gondolom) je napravljena tako da što bolje drži crtaći instrument u sjecištu veza. Time se može zanemariti greška pri crtanju malenih detalja jer inače se pisača glava naginje u stranu, što pomiče crtaći instrument u odnosu na veze te tako uvodi grešku u obliku distorzije slike. U ovom radu je napravljena pisača glava korištenjem kugličnih ležajeva i pleksiglas plastike za konstrukciju kao što je prikazano na slici 3.1. Koristimo tri kuglična ležaja kao što su prikazana na slici 3.2. montirane na bakrenu cijev (jer bakar je mekši od čelika, te ne može značajno oštetiti ležajeve u slučaju nezgode). Prvi ležaj je statičan i služi samo kao dodatna težina i kao spojna površina za plastičnu bazu koja će se spojiti. Ostala dva ležaja će se preko plastičnih spojnica spojiti na vodove, te time smo omogućili jednostavno centriranje crtaćeg

instrumenta, samo se montira u cijev (naprimjer preko dva vijka koji osiguravaju da neće klizati crtaći instrument pri radu). Plastična baza pisače glave služi kao držač za servo motor koji podiže i spušta pisaču glavu, te time odvaja crtaći instrument, te služi kao stabilizator u slučaju da se pisača glava želi prevrnuti u stranu.



Slika 3.1. Sastavljena pisača glava

Slika 3.2. Bakrena cijev sa tri kuglična ležaja

3.1.1. Problem kod fizičkog modela

Fizički model ima pisaču glavu koja visi sa dvije veze pomoći sile teže. Pri crtanju crtaći instrument ostavlja trag na crtaćoj površini. Pri tome nastaje trenje između instrumenta i površine. To trenje može prouzrokovati problem pri radu stroja u obliku trzanja. Nastanak problema možemo povezati sa napetošću vodova, jer manje napet vod je više osjetljiv na sve pomake pa pri oštroj kretnji skače od površine pa na taj način trza.

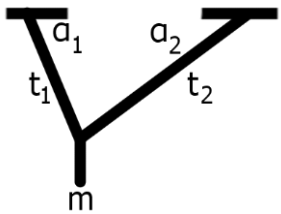
Da se problem lakše ilustrira, zadaju se neki početni uvjeti [10]:

- masa pisače glave iznosi *m*, te traži se da je napetost vodova između *m*/2 i 1.5*m*
- uvodi se i problem rezolucije gdje promjenom koordinatnog sustava iz kartezijevog u alternativni dovodi do nejednolike promjene u rezoluciji koraka, što dovodi do područja gdje je rezolucija koraka pregruba

Preko računalnog programa pisanog u Python-u (originalni autor koda je Bill Ola Rassmussen) može se dobiti dvodimenzionalni pregled koji prikazuje sve pozicije crtaće površine obojane na takav način da prikazuje mjesta gdje je napetost neke veze preniska (ispod 0.5*m*) ili previsoka(iznad 1.5*m*) i da prikazuje mjesta gdje je rezolucija ispisa niska (ako jedinični pomak u alternativnom sustavu uzrokuje više nego 1.4 puta veći pomak u kartezijevom sustavu onda se to područje označi kao nepovoljno).

Zbog implementacije se moraju prvo poznavati potrebni matematički izrazi.

Proračun napetosti vodova se računa u situaciji da imamo uteg mase *m* obješeno preko dva voda t_1 i t_2 od kojih svaki ima svoj kut α_1 i α_2 kao što bi bilo na slici 3.3.



Slika 3.3. Prikaz sustava sila

Ako su sile u ravnoteži dobiju se sljedeći izrazi:

Horizontalne sile:

$$t_1 \cdot \cos\alpha_1 = t_2 \cdot \cos\alpha_2$$

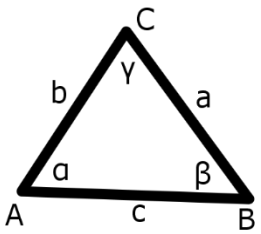
Vertikalne sile

$t_1 \cdot \sin\alpha_1 + t_2 \cdot \sin\alpha_2 = m$

Ako se želi izraziti napetost dobije se:

$$t_1 = \frac{\cos\alpha_2 \cdot m}{\cos\alpha_1 \cdot \sin\alpha_2 + \sin\alpha_1 \cdot \cos\alpha_2}\ [N]$$
$$t_2 = \frac{\cos\alpha_1 \cdot m}{\cos\alpha_1 \cdot \sin\alpha_2 + \sin\alpha_1 \cdot \cos\alpha_2}\ [N] \tag{3-1}$$

Proračun za rezoluciju se dobije preko kosinusnog poučka prikazanog na slici 3.4:



Slika 3.4. Prikaz kosinusnog poučka.

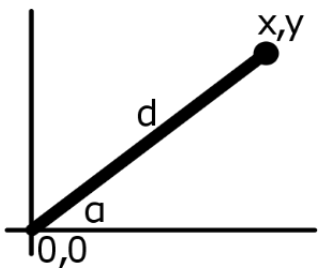
Osnovni izraz za dužinu stranice preko kosinusnog poučka je:

$$a^2 = b^2 + c^2 - 2 \cdot b \cdot c \cdot \cos\alpha$$

Time je izraz za kut:

$$\alpha = \arccos\frac{b^2 + c^2 - a^2}{2 \cdot b \cdot c} \tag{3-2}$$

Konačno treba izraziti veze između kartezijevog i alternativnog koordinatnog sustava kao što je prikazano na slici 3.5.



Slika 3.5. Prikaz za pretvaranje sustava

Potrebna su pretvaranja između dva skupa:

- iz kartezijevog sustava u polarni oblik
- iz polarnog oblika natrag u kartezijev sustav

Dobiju se sljedeće formule:

$$x, y = d \cdot \cos\alpha\,, d \cdot \sin\alpha \;\; [m, m]$$
$$d, \alpha = \sqrt{x^2 + y^2}, \mathrm{atan2}(y, x) \;\; [m, °] \tag{3-3}$$

Sa ova tri seta jednadžbi (3-1, 3-2, 3-3) imamo sve izraze potrebne za program [10] (cijeli kod programa se nalazi u prilogu B).

Na početku programa imamo konstantne varijable za širinu i visinu stroja. Zato što su u programu sve dužine jedinične, može se aproksimirati tako da se unesu testne dimenzije u milimetrima. Dalje u funkcijama se nalaze opsezi za testiranje napetosti i rezolucije koje se mogu mijenjati ovisno o zadanim granicama.

Program radi tako da za svaki piksel na slici provjeri dva uvjeta, napetost i rezoluciju kao što je prikazano u isječku koda 3.1.

```
def calc_pixel(draw, p):
    tension_check = tension_ok(p)
    resolution_check = resolution_ok(p)
    if not tension_check and not resolution_check:
        draw.point(p, "#2620d5")
    if not tension_check and resolution_check:
        draw.point(p, "#4876FF")
    if tension_check and not resolution_check:
        draw.point(p, "#FF7F24")
    # default to background color
```

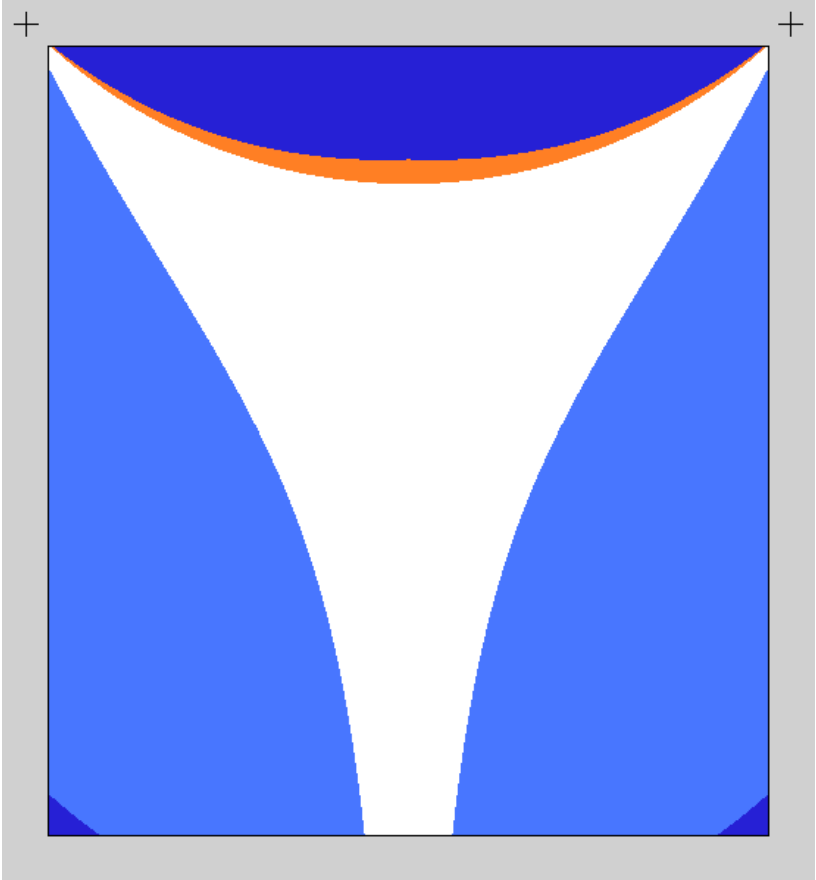
Isječak programa 3.1. Funkcija za određivanje boje piksela

Što daje sljedeću tablicu mogućih rezultata 3.1.:

Tablica 3.1. Mogući rezultati funkcije za određivanje boje piksela

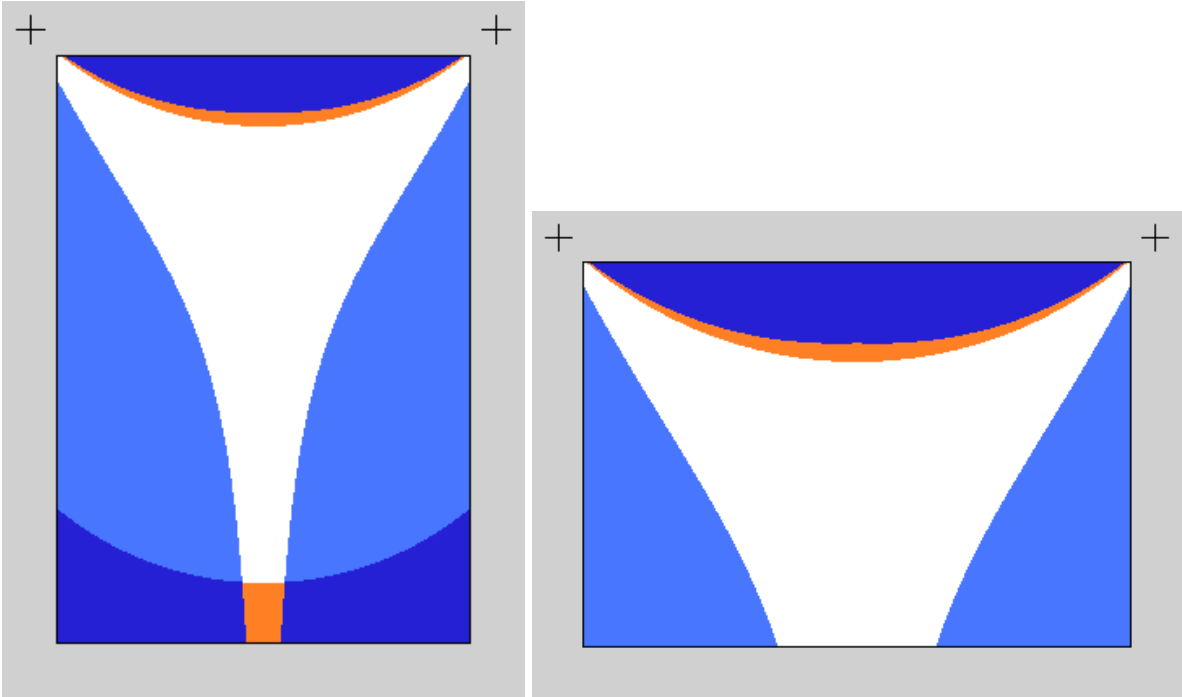
Napetost	Bit	Rezolucija	Bit	Boja	
Van opsega	0	Van opsega	0	Tamno plava	#2620D5
Van opsega	0	U opsegu	1	Svijetlo plava	#4876FF
U opsegu	1	Van opsega	0	Narančasta	#FF7F24
U opsegu	1	U opsegu	1	Bijela	#FFFFFF

Ako unesemo podatke prototipa (širina: 553, visina: 600) dobijemo rezultat kao na slici 3.6.:



Slika 3.6. Rezultat programa sa podatcima prototipa

Vidljivo je da postoji u sredini stroja veliko područje sa optimalnom napetošću veza i rezolucijom da ne bi trebalo biti problema tijekom rada ako se nalazi u označenom bijelom području. Za generalne slučajeve vrijedi da omjer četiri naprema tri u smjeru crtanja daje zadovoljavajuće rezultate (4:3 visina i širina ako se crta uspravno, 3:4 visina i širina ako se crta položeno kao što je vidljivo na slici 3.7.).



(a) (b)

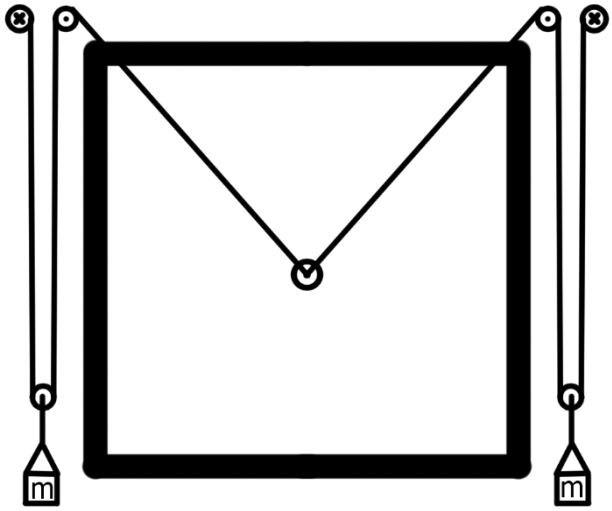
Slika 3.7. Rezultat programa pri omjeru 4:3 visine i širine (a);
Rezultat programa pri omjeru 3:4 visine i širine (b)

Općenito se mora paziti da crtaća površina nije preniska, jer u gornjem dijelu (tamnoplava boja) je previše napeto i loša je rezolucija. Ne bi se trebalo crtati u području ispod motora (svijetlo plava) jer se može dogoditi da motor počne visjeti o samo jednu vezu zbog nedostatka napetosti na suprotnoj vezi, pa zbog trenja crtaćeg instrumenta krene trzati tijekom rada.

Još mogućih problema na koje je potrebno paziti kad je veličina stroja velika (par metara u svaku stranu) [11]:

- brzina je ista koliko god velik stroj bio (osim ako se ne promijeni oblik zupčanika koji motori pogone), te ako je crtež dovoljno detaljan može trajati danima da ga stroj nacrta
- kvaliteta ispisa je najčešće u redu jer je „bijela površina“ koja je pogodna za crtanje veća kako se stroj povećava, ali onda dolazi do drugog problema, težina vodova. Vodovi se mogu početi savijati pod svojom težinom te time uvode distorzije u crtež
    - moguće je do neke granice eliminirati ako se poveća težina pisače glave
- greške pri crtanju mogu biti skupocjene jer se mora cijela crtaća površina očistiti ili promijeniti ako je od papira, te gubi se uloženo vrijeme
    - ovakve greške se eliminiraju tako da se pomno pregledaju svi faktori prije početka crtanja (provjeriti dali je crtaći instrument ipsravan, dali je zadani crtež pravilno postavljen, itd.)

- ponekad je poželjno smanjiti detalje, ali to nije izvedivo ako se koriste markeri jer idu samo do neke najveće veličine, moguće je onda umjesto markera koristiti boju u spreju
- orijentacija stroja postaje problematična jer nije teško ići u širinu, ali teško je ići u visinu
    - vodovi bi trebali biti dovoljno dugački da se mogu dijagonalno postaviti na stroju (moraju biti barem 1.4 ($\sqrt{2}$) puta veći od ukupne visine stroja), te dolazi do problema kad vodovi moraju visjeti sa strana stroja a da protuteže ne diraju pod
    - moguće rješenje je da se vrh stroja nalazi 1.4 puta visine stroja visoko postavljeno
    - ovaj problem kod većih strojeva se može riješiti koristeći sustav kolotura (primjer na slici 3.8.) kojima se smanjuje mjesto potrebno za držanje viška vodova (onda se mora namjestiti težina protuteža jer će one iziskivati manje sile na vodove koje vode do motora)



Slika 3.8. Primjer sustava za skraćivanje mjesta potrebnog za vodove koristeći koloture

3.2. Upravljački program

Upravljački program pod nazivom *polargraphcontroller* služi kao glavna veza sa Arduino pločicom. Pisano je u *Processing* [12] *IDE*-u (eng. *Intelligent Development Enviroment*, Inteligentno Razvojno Okruženje, IRO), što je razvojno okruženje koje koristi otvoreni izvorni kod, te primarno služi za razvoj aplikacija vezano za vizualni dizajn koristeći razne biblioteke za obradu slika u Java programskom jeziku [13].

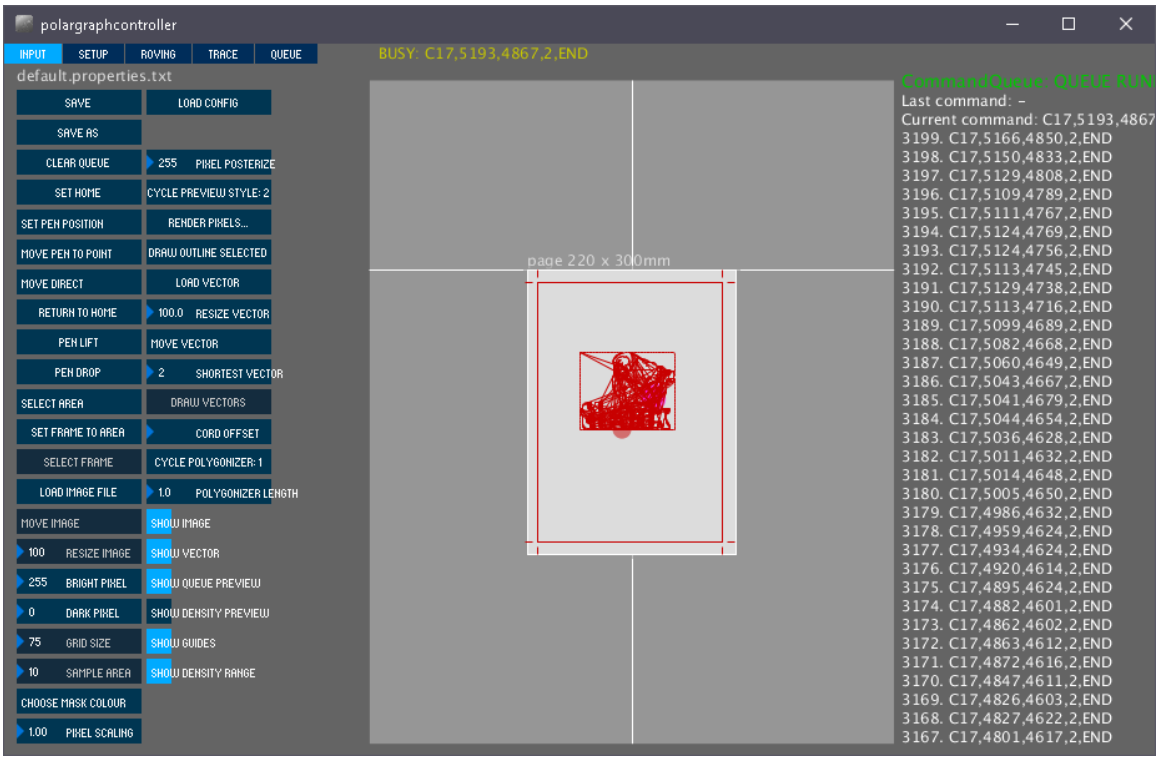
Postoje dva načina instalacije upravljačkog programa:

- može se skinuti gotov kompiliran program koji bi trebao općenito raditi
- ako ne radi prvi način zbog mogućih nekompatibilnosti moraju se pratiti upute na internetu za ručno kompiliranje izvornog koda u *Processing*-u

Sam upravljački program je dizajniran da radi sa ekranima osjetljivim na dodir, svi klizaći mijenjaju vrijednost tako da se drži klik/dodir na njih pa pomicanjem gore ili dolje se mijenja vrijednost klizaća.

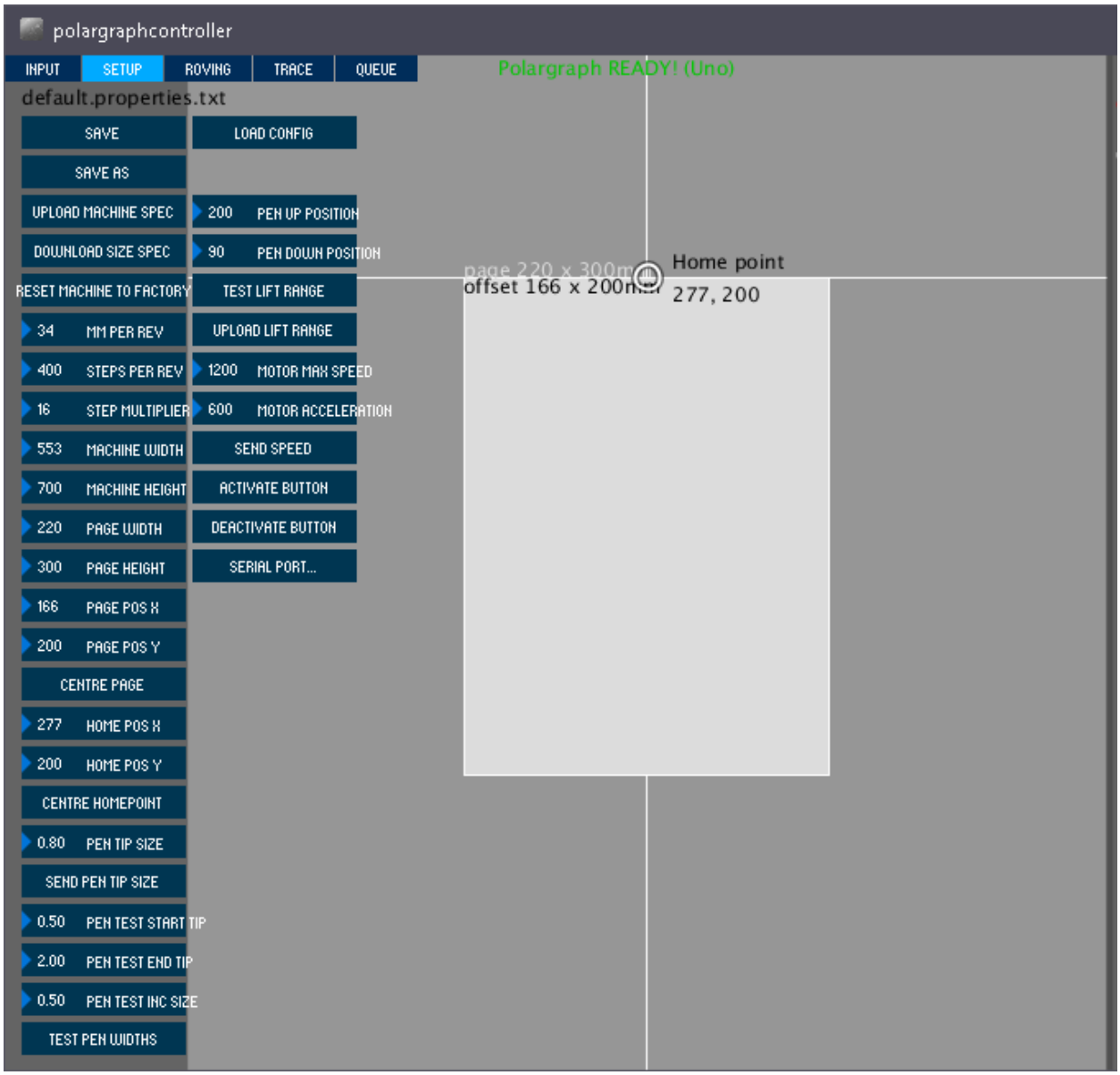
Upravljački program se sastoji od jednog prozora koji ima više kartica:

- kartica „*Input*“ (ulaz) koja je prikazana na slici 3.9. služi kao glavna kartica upravljačkog programa. U njoj se učitavaju datoteke za ispis, te pruža grafičko sučelje trenutnog stanja stroja pomoću kojeg se namješta ispis i upravlja stroj preko naredbi
    - potrebno je prije crtanja postaviti okvir crtanja, to se učini tako da se odabere „*Select area*“ (odaberi područje) i označi površina na stranici(svijetlo siva boja), nakon odabiranja kliknuti na „*Set frame to area*“ (postaviti okvir na područje) da bi se odredio okvir crtanja
        - dio vektora ili slike van ovog područja će se zanemariti i neće dobiti vezanu naredbu
        - ova opcija se koristi i kod traganja slike u kartici „*Trace*“ te ona određuje koji dio slike se traga
    - moguće je crtati učitane vektorske datoteke ili slike koje upravljački program obradi i prevede u stroju razumljiv oblik, kao što je prikazano u slici A.1.



Slika 3.9. Prikaz „*Input*“ (ulaz) kartice

- kartica „*Setup*“ (postavke), koja je prikazana u slici 3.10., služi za postavljanje svih potrebnih podataka fizičkog modela (kao što je udaljenost između motora, specifikacije motora, …)
    - većina postavki se za neki sustav moraju jednom mjeriti ili izračunati pa će dalje Arduino ploča i računalo zapamtiti podatke, ali mora se spremiti ručno jer program ne sprema automatski klikom/dodirom na „*Save*“ (spremi)
    - pri svakom paljenju upravljačkog programa, ono ponovno šalje naredbe za brzinu i akceleraciju motora i debljinu crtaćeg instrumenta jer ih Arduino pločica ne sprema u svoju trajnu memoriju



Slika 3.10. – Prikaz „*Setup*“ (postavke) kartice

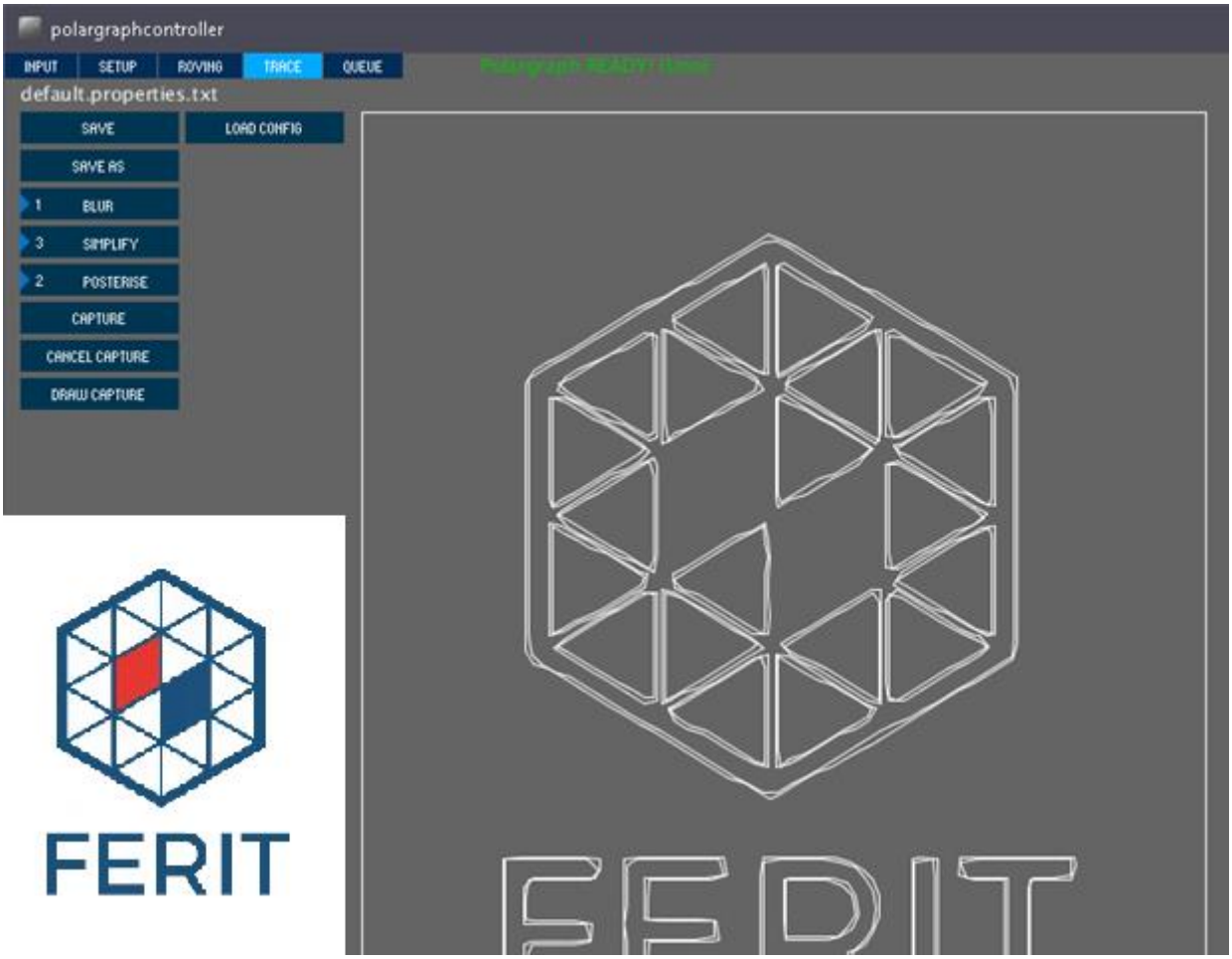
- kartica „*Roving*“ (lutanje) koja je prikazana na slici 3.11. se može koristiti samo sa *Polarshield* dodatkom za *Espressif ESP-32* mikroupravljače jer koristi specijalne naredbe koje nisu dostupne na Arduino verziji zbog kompleksnosti
    - omogućuje ispis teksta po zadanom području koristeći ugrađenu vektorsku datoteku koja sadrži englesku abecedu u vektorskom obliku, što omogućava znatno brži ispis slova i brojeva nego da se prevede standardna abeceda u vektorsku ili raster datoteku
    - omogućuje mogućnost crtanja „Norveškog piksela“ prikazanog na slici A.2., to je tip crtanja gdje se zadana slika pretvori u poseban tip naredbe s kojom stroj pravi koncentrične kružnice varirajućih debljina, debljina kružnice u određenom mjestu

podudara se sa svjetlošću piksela u istoj lokaciji (tamniji piksel daje deblju kružnicu u istom mjestu), daje sličan rezultat kao obrada slike u kontroleru za Arduino verziju

Slika 3.11. – Prikaz „*Roving*“ (lutanje) kartice

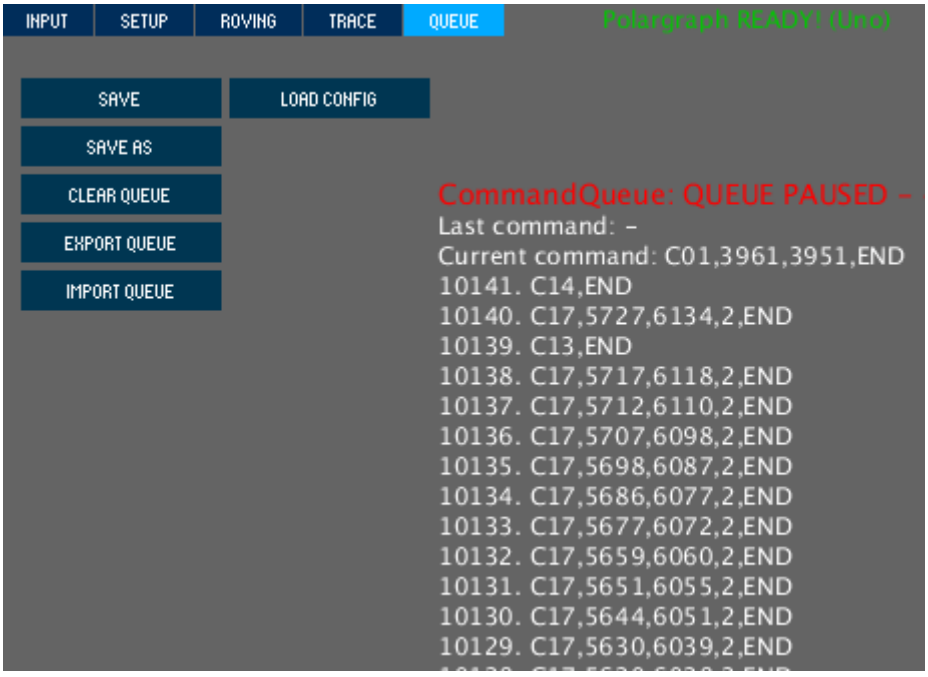
- kartica „*Trace*“ (trag, trasirati) koja je prikazana na slici 3.12. služi za grubo traganje neke slike koja se nalazi u okviru crtanja u vektorski oblik, sadrži više klizača pomoću kojih se postavlja konačni izgled vektora (mogu se namještati za preciznost ili za, naprimjer, artističko izražavanje)
    - klizač „*Blur*“ (mutno) regulira koliko fino program traga sliku, manje vrijednosti vjernije tragaju rubove boja dok veće vrijednosti daju grublje ali mekše traganje(efektivno smanjuje buku/grešku u tragu ali žrtvuje detalje)
    - klizač „*Simplify*“ (pojednostavi) regulira koliko program pojednostavljuje trag tako da izglađuje oštre prijelaze

- klizač „*Posterize*“(Posterizacija) regulira koliko sveukupno tonova boje program prati tijekom traganja slike, manje vrijednosti daju manje sveukupno boja pa je slika manje detaljna, veće vrijednosti prate više boja pa je slika više detaljna



Slika 3.12. Prikaz „*Trace*“ (trag, trasirati) kartice, u doljnjem lijevom kutu slike se nalazi originalna slika koju program traga

- kartica „*Queue*“ (red) koja je prikazana u slici 3.13. služi za snimanje generiranog popisa naredbi u tekstualnoj datoteci za buduću uporabu, te u istoj kartici možemo učitati te datoteke
    - spremljene naredbe pamte pozicije koje su dobile pri generiranju, ne mogu se micati naknadno
    - ova kartica se može koristiti sa *Polarshield* dodatkom tako da se usnimi generirani kod na SD kartici koja se može koristiti sa navedenim dodatkom tako da ono bez računala izvršava naredbe spremljene na SD kartici



Slika 3.13. Prikaz „*Queue*“ (red) kartice

3.3. Način funkcioniranja upravljačkog programa

Upravljački program funkcionira tako da šalje naredbe Arduino pločici koja sadrži *Polargraph* softver preko USB UART serijske veze. Ploča pri dovršavanju trenutne naredbe javlja natrag kada je spremna primiti sljedeću naredbu. Preko te naredbe za javljanje natrag upravljački program može prepoznati o kakvom se uređaju radi, jer ima tri vrste poruka za javiti: „READY“ za standardne modele *Polargraph*-a koje rade na Arduino Uno-u, „READY_100“ za modele koje koriste Arduino Mega ploče te „READY_200“ za modele koje koriste *Espressif EPS-32* ploče i Polarshield dodatak. Može se trenutno stanje Arduina vidjeti gore lijevo u glavnom prozoru upravljačkog programa u obliku „*Polargraph READY! (*model*)*“ kada je spreman i „*BUSY: *kod naredbe**“ kada je naredba u tijeku. Ovim načinom je uspostavljena jednostavna ali pouzdana komunikacija.

Trenutno postoji 19 naredbi, od kojih 9 su naredbe za postavljanje postavki stroja [14], popis naredbi se nalazi u tablici 3.2.

(Dijelovi naredbi u uglatim zagradama su neobavezni te mogu se izostaviti)

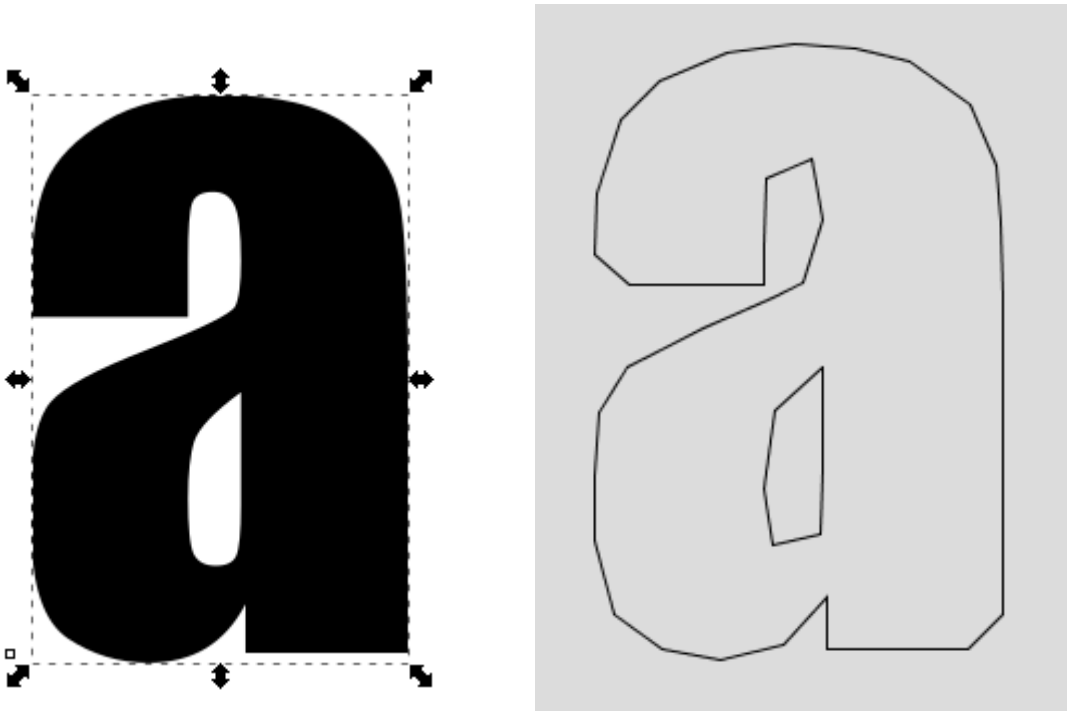
Tablica 3.2. Naredbe Polargraph sustava

Naredba	**Opis naredbe**
C01, <lijeva dužina>,<desna dužina>,END	Pokreni motore dok nisu došli do zadane dužine mjerene u koracima motora
C02,<debljina crtaćeg instrumenta>,END	Postavlja debljinu crtaćeg instrumenta na zadanu vrijednost u milimetrima, prima decimale
C05,<lijeva dužina>,<desna dužina>,<veličina>,<svjetlina>,END	Nacrtaj piksel koristeći osjenčavanje valovima na lokaciji danom dužinama sa danom veličinom strana i svjetlinom 0-255
C06,<lijeva dužina>,<desna dužina>,<veličina>,<svjetlina>,END	Slično kao gornja naredba ali osjenčavanje se provodi nasumičnim potezima
C07,<način određivanja smjera>,<smjer>,END	Mijenja smjer crtanja pri crtanju piksela, način određivanja govori kako da odredi smjer, te smjer se odnosi na orijentaciju kretanja: prema ili od kojeg motora
C11,<veličina piksela>,<početna osjenčanosti>,<konačna	Naredba za crtanje test polja piksela od najmanje osjenčanosti piksela pa inkrementalno do konačne osjenčanosti piksela

osjenčanost>,<inkrementalni korak osjenčanosti>,END	
C09,<lijeva dužina>,<desna dužina>,END	Govori stroju gdje se trenutno nalazi pisača glava u udaljenostima od oba motora.
C14,[<pozicija servo motora>],END	Spušta crtaći instrument, ima moguće polje da kaže koliko da se pomakne servo motor, te se ne mora pozivati ta brojka sljedeći puta. Za trajno spremanje koliko da se pomakne koristi se naredba C45
C13,[<pozicija servo motora>],END	Diže crtaći instrument, vrijedi isto kao kod prijašnje naredbe
C45,<donja pozicija>,<gornja pozicija>,[1],END	Trajno sprema pozicije ako je prisutna [1] u liniji, inače napravi test pomak naprijed nazad između vrijednosti
C17,<lijeva konačna dužina>,<desna konačna dužina>,<najmanji linijski segment>,END	Najosnovnija naredba za crtanje, pomiče pisaču glavu prema konačnoj udaljenosti od motora. Pritom uzima u obzir da svaki pomak rastavi u red manjih pomaka zbog glatkoće ispisa. Koliko su mali segmenti određuje zadana varijabla 'najmanji linijski segment'
C24,<širina pisaćeg stroja>,<visina pisaćeg stroja>,END	Postavlja veličinu stroja zbog potrebnih trigonometrijskih kalkulacija u Arduino ploči
C26,END	Arduino ploča vraća sve spremljene podatke o stroju
C27,END	Resetira sve postavke na tvornički zadane
C29,<mm po okretaju>,END	Šalje ploči koliko se promijeni duljina jedne veze pri punom okretaju motora
C30,<koraka po okretaju>,END	Šalje ploči koliko koraka motora je potrebno za pun okretaj
C31,<najveća brzina motora>,END	Šalje ploči kolika je najveća dozvoljena brzina motora
C32,<ubrzanje motora >,END	Šalje ploči koliko je ubrzanje motora
C37,<korak mikrokoračenja>,END	Šalje ploči koliko je postavljeno mikrokoračenje motora

Upravljački program radi sa vektorima tako da učitanu datoteku obradi te izvuče sve elemente. Nakon toga sve nelinearne elemente aproksimira linearnim, naprimjer kružnicu će aproksimirati sa brojem ravnih linija koje prate konturu originalne kružnice, elemente sa isječenim konturama

će aproksimirati tako da prati svaku konturu, naprimjer ako je zadana kružnica sa unutrašnjim izrezom (naprimjer još jednom kružnicom da se dobije oblik prstena) program to obrađuje kao dvije konture te ih onda obje aproksimira s brojem ravnih linija. Program ne uzima u obzir zadanu debljinu linije pri crtanju (može biti 0 i upravljač će ju i dalje nacrtati).



(a) (b)

Slika 3.14. Izgled objekta u vektorskom uređivaču (a);
Izgled objekta u upravljačkom programu (b)

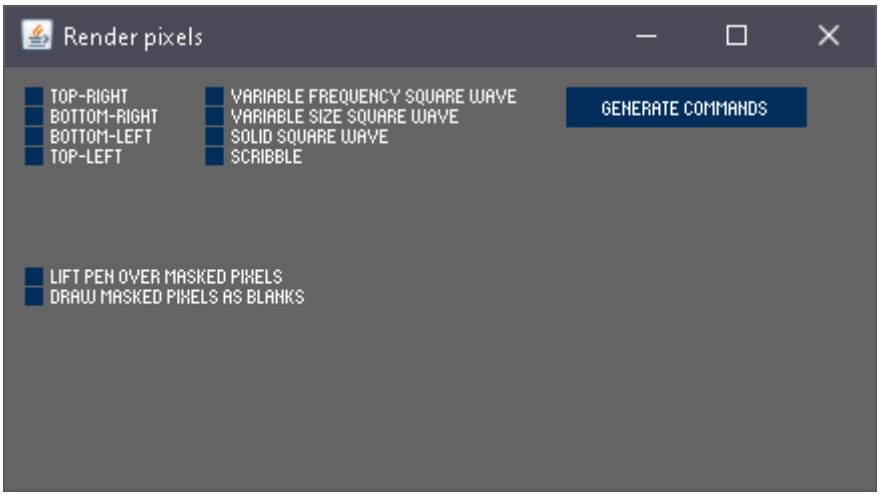
Kao što se vidi na slici 3.14. za slovo „a“, svi obrubi slova su pretvoreni u linijske segmente. Svaki linijski segment stvoren na taj način dobije svoju naredbu u popisu naredbi. Na trenutnom primjeru slova „a“ bi dobili 2 skupa naredbi, prvi skup za vanjski obrub slova, te drugi skup za šupljinu slova „a“. Ta dva skupa bi odvojile naredbe za dizanje, brzi pomak i spuštanje crtaćeg instrumenta u točnoj poziciji. Ako postoji više objekata u jednoj vektor datoteci, upravljački program ide kroz sve objekte redom kako su navedeni u datoteci. Pritom između naredbi za crtanje svakog objekta ubaci dizanje, brzi pomak i spuštanje crtaćeg instrumenta. To nastavlja dok ne obradi sve objekte vektorske datoteke. Na kraju upravljački program ispiše sve naredbe u jednom stupu koji predstavlja tok naredbi koji će se slati mikroupravljaču.

Brzina ispisa crteža ovisi o postavkama zadanim u „*Setup*“ i „*Input*“ karticama. Sa većom maksimalnom brzinom i većom akceleracijom, motori mogu brže crtati, ali to dolazi najčešće sa mogućim problemima trzanja pri visokim brzinama te isto je moguće da motori zastanu zbog previsokog broja koraka koje upravljačka komponenta želi da motori izvedu te tada samo trzaju na mjestu. Nakon toga se učitava vektor datoteka. U „*Input*“ kartici se vidi pregled učitane vektor

slike te postoji klizač s imenom „*Shortest vector*“, ovaj klizač određuje kolike dužine će biti najmanji linijski segment pri izradi naredbi. Ako se postavi na 0, onda će upravljački program izraditi sve naredbe tako da što bliže i točnije prati konturu. Ako se povećava vrijednost klizača, onda upravljački program izrađuje naredbe tako se da svi linijski segmenti koji su kraći od vrijednosti klizača (izraženo u broju koraka motora) aproksimiraju zajedno sa nekim drugim linijskim segmentom, te time spaja dvije linije u jednu. Ovim postupkom se gubi kvaliteta detalja crteža ali se smanjuje konačni broj naredbi te time je vrijeme ispisa crteža kraći. Kod crteža koji koriste jako puno ravnih linija ovaj postupak će imati minimalan utjecaj na kvalitetu konačne slike, dok kod crteža koje sadrže puno zaobljenih kontura i detalja će više utjecati na kvalitetu konačne slike.

Kada upravljački program učita sliku koja nije vektor, ono ju ne može odmah nacrtati, nego ju mora pretvoriti u razumljiv oblik. To je moguće u „*Input*“ kartici učiniti sljedećim koracima (rezultat je sličan „norveškom pikselu“ iz „*Roving*“ kartice, ali ovo je postupak za Arduino verziju stroja):

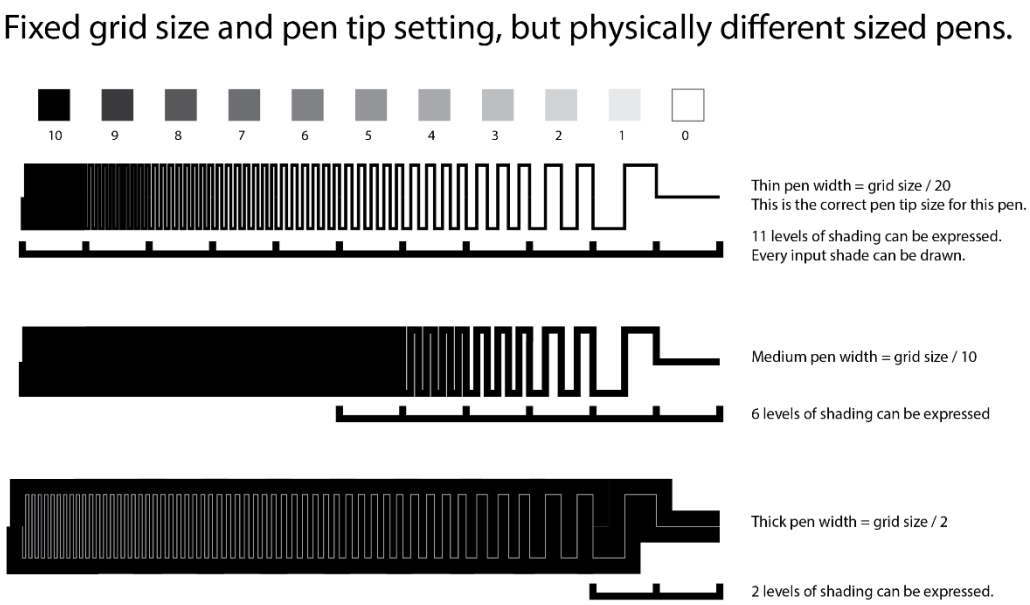
1. Učitava se željena slika
2. Odabire se područje pomoću „*Select area*“ (odaberi područje) naredbe
3. Pomoću klizača „*Brigth pixel*“ (svijetli piksel) i „*Dark pixel*“ (tamni piksel) označavamo gornju i donju granicu za svjetlinu piksela (ono govori koliko će gusto nacrtati linije ovisno o svjetlini piksela u istom mjestu)
4. Pomoću klizača „*Grid size*“ (veličina rešetke) namještamo koliko će svaki piksel biti velik, što je manja vrijednost klizača, to je manja rezolucija opsega nijansi za svaki nacrtani piksel ali je slika detaljnija
5. Pomoću klizača „*Sample size*“ (prostor uzorkovanja) mijenjamo veličinu područja uzorkovanja u obliku kružnice pomoću koje se aproksimira nijansa piksela (veće vrijednosti)
6. Naredbom „*Render pixels*“ (prikaži piksele) čiji je dijalog prikazan na slici 3.15. dobijemo dodatni dijalog pomoću kojeg se namješta konačni ispis koristeći dane opcije za mjesto izvora koncentričnih kružnica piksela i načinom crtanja tih piksela



Slika 3.15. Prikaz „Render pixels“ (prikaži piksele) dijaloga

7. Klikom na „*Generate commands*“ (generiraj naredbe) dobijemo u popisu naredbi sve naredbe za svaki piksel

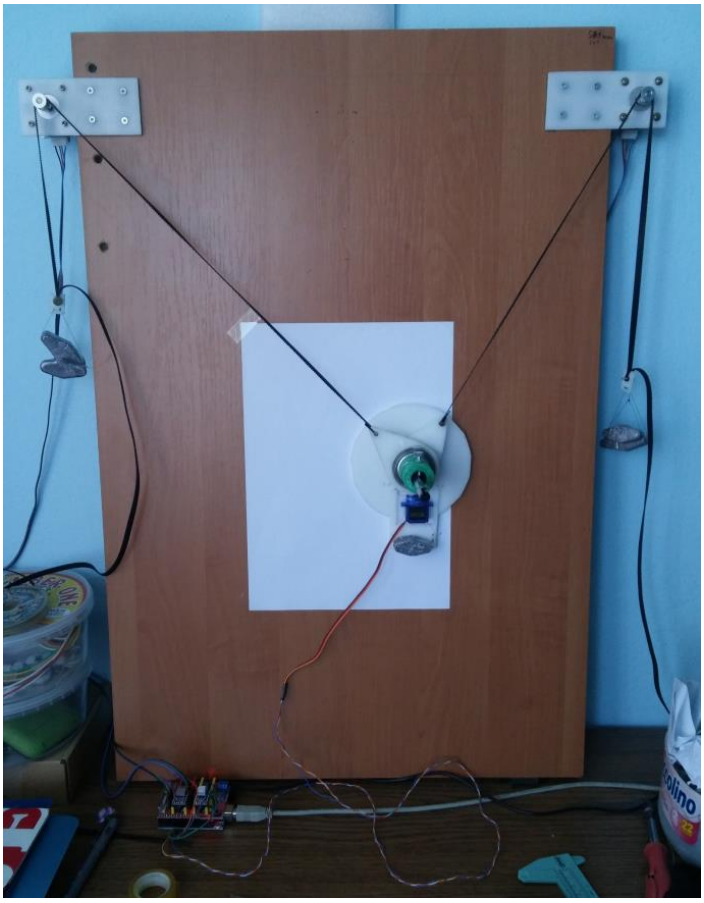
Potrebno je napomenuti da svaki generirani „piksel“ koji crta ovako ovisi o debljini crtaćeg instrumenta i veličine rešetke [15]. Ako je debljina prevelika a rešetka premalena onda će moći izraziti samo dvije nijanse. Što je veći raspon rešetke i debljine instrumenta onda je dostupan veći opseg nijansi pri crtanju kao što je prikazano na slici 3.16.



Slika 3.16. Prikaz korelacije debljine crtaćeg instrumenta i veličine rešetke

3.4. Realizacija i rezultati fizičkog modela

Sagrađeni fizički model je montiran na drvenoj ploči grube teksture zbog potreba lijepljenja papira na površinu kao što je prikazano na slici 3.17.

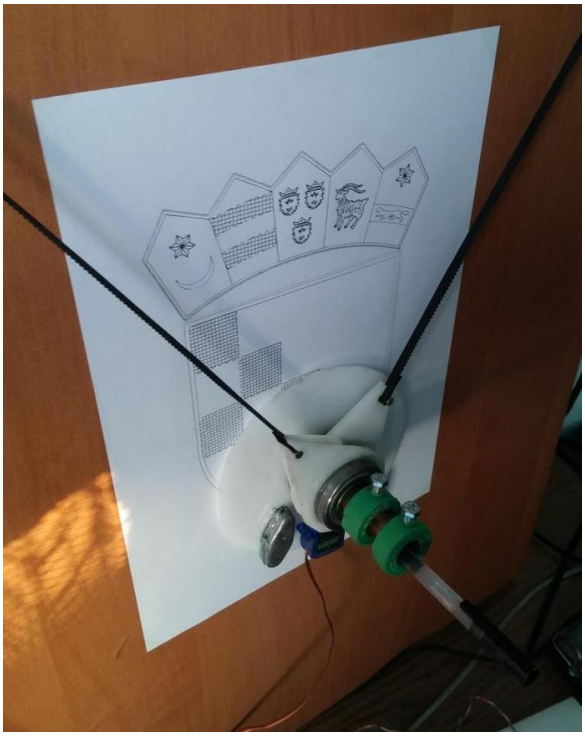


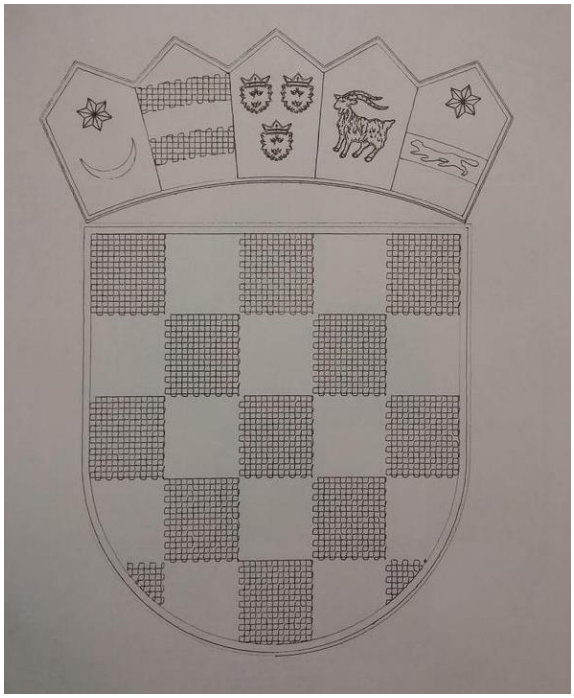
Slika 3.17. Fizički model

Par koračnih motora su montiran na gornjim kutovima drvene ploče tako da su si međusobno vodoravni po visini, inače se mogu uvesti distorzije ili rotacije u crtež. Zato što su motori pomaknuti u stranu koristeći pleksiglas pločice veće je područje crtanja na samoj ploči jer je područje premale napetosti pomaknuto. Na samoj ploči je označen položaj gdje se nalazi početna točka sustava crtanja za jednostavno postavljanje, papir bi se postavio tako da se sredina gornjeg ruba papira nalazi na toj točki. Time se postiže udaljavanje od gornjeg dijela modela gdje je napetost i rezolucija loša kao što je objašnjeno u podpoglavlju 3.1.1.

Fizički model je dao zadovoljavajuće crteže sa normalnom brzinom i akceleracijom te dao je lagano izobličene crteže pri visokim brzinama i akceleracijama zbog trzanja pisače glave pri radu. Na slici 3.18. se vidi rad sa normalnom brzinom, te nema uočljivih izobličenja, dok kod slike 3.19. se povećala brzina i akceleracija pri crtanju zbog skraćivanja vremena crtanja te mogu se uočiti

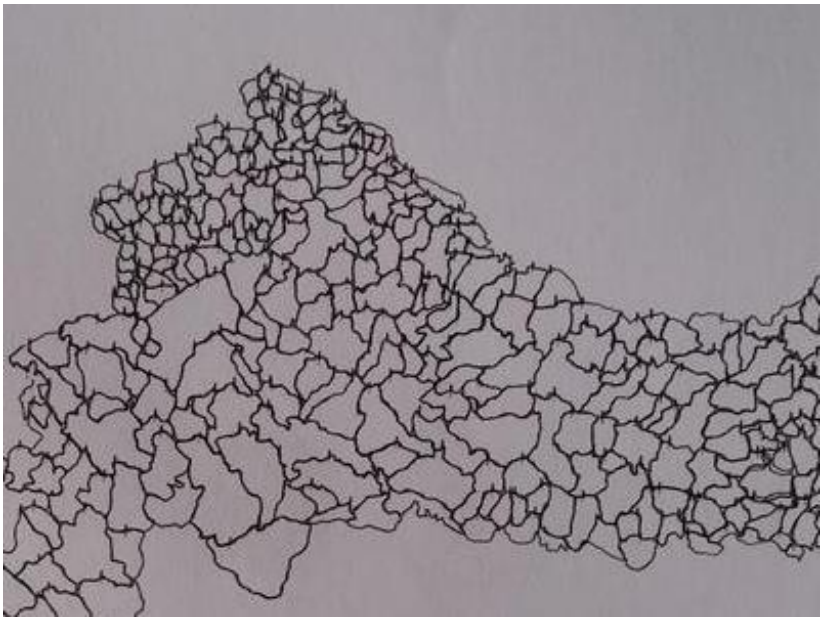
greške u obliku kratkih vertikalnih crta koje nastaju zbog trzanja pisače glave te zbog trzanja servo motora za podizanje i spuštanje pisače glave sa površine.





(a) (b)

Slika 3.18. Model pri crtanju hrvatskog grba (a);
Gotov crtež hrvatskog grba (b)



Slika 3.19. Isječak crteža hrvatskih općina

Cijeli album slika se nalazi na poveznici u literaturi [16]. Popis video snimaka koji prikazuju ubrzanu snimku modela se nalaze na poveznici u literaturi [17].

4. ZAKLJUČAK

U ovom radu je izrađen model koji koristi *Polargraph* softver baziran na biangularnom sustavu za crtanje po velikim površinama. Model može biti u više dimenzija, tek toliko da crta na A4 papiru ili može biti par metara u dijagonali tako da crta po ploči ili zidu. Pisača glava se može sastaviti na više raznih načina, od par žica i uredskih materijala do dizajna posvećenim ovakvom tipu stroja koje mogu koristiti razne vrste crtaćih instrumenata. Crtaća površina modela na kojoj je sve montirano je od drveta zbog mogućnosti lijepljenja papira za crteže. Model nije teško mijenjati jer sve što se mora napraviti je montirati motore na drugoj lokaciji i ponovno izmjeriti duljinu između motora za postavke. Za montiranje na školskoj ploči uzima se u obzir da nema dovoljno mjesta sa strane ploča za motore, zbog toga se moraju montirati na takav način da im vratila motora pokazuju prema ploči.

LITERATURA

[1] G.B.M. Zerr, Biangular coordinates, Taylor & Francis, Ltd., veljača 1910.

[2] S. Noble, 'Who?', Polargraph, dostupno na: http://www.polargraph.co.uk/who/

[3] S. Noble, 'What's a polargraph?', Polargraph, dostupno na: http://www.polargraph.co.uk/whats-a-polargraph/

[4] Arduino, 'About us', dostupno na: https://www.arduino.cc/en/Main/AboutUs

[5] B. Kruger, 'Arduino CNC shield', Protoneer, dostupno na: https://blog.protoneer.co.nz/arduino-cnc-shield/

[6] Pololu, 'DRV8825 Stepper Motor Driver Carrier, High Current', dostupno na: https://www.pololu.com/product/2133

[7] Guido van Rossum, 'The History of Python', blogspot, dostupno na: http://python-history.blogspot.com/

[8] ByMisan, 'Different drive modes for a unipolar stepper motor.', Wikipedia Commons, dostupno na: https://commons.wikimedia.org/w/index.php?curid=9787501

[9] D. Collins, 'What is microstepping?', Linear Motion Tips, dostupno na: https://www.linearmotiontips.com/microstepping-basics/

[10] B. Olla Rasmussen, 'V plotter design', 2e5.com, dostupno na: http://2e5.com/plotter/V/design/

[11] S. Noble, 'Large Machines', GitHub, dostupno na: https://github.com/euphy/polargraph/wiki/Large-Machines

[12] S. Noble, 'euphy/polargraphcontroller', GitHub, dostupno na: https://github.com/euphy/polargraphcontroller

[13] J. Meyer, 'Programming 101: The How and Why of Programming Revealed Using the Processing Programming Language', stranica 121. nadalje, Apress, dostupno na: https://books.google.hr/books?id=yypgDwAAQBAJ&pg=PA121&redir_esc=y#v=onepage&q&f=false

[14] S. Noble, 'Polargraph machine commands and responses', GitHub, dostupno na: https://github.com/euphy/polargraph/wiki/Polargraph-machine-commands-and-responses

[15] euphy, ‘Empty pixels’, GitHub, dostupno na:
https://github.com/euphy/polargraph/wiki/Empty-pixels

[16] A.Tufeković, ‘Završni rad – slike (Anto Tufeković)’, Imgur, dostupno na:
https://imgur.com/a/JfF2nhs

[17] A.Tufeković, ‘Polargraph – Anto Tufeković’, Youtube, dostupno na:
https://www.youtube.com/playlist?list=PL06Hc9F-aTqju6pyeWmlj3FKajodqMDq1

SAŽETAK

Naslov: Automat za crtanje po školskoj ploči

U ovom radu je izrađen model za crtanje koji koristi biangularni sustav umjesto kartezijevog koordinatnog sustava. Arduino ploča prima naredbe od računala te izvršava te naredbe preko koračnih motora. Bit ovog rada je da se model može lagano prilagoditi crtaćoj površini (velikoj, maloj, glatkoj površini kao bijele ploče, gruboj površini za lijepljenje papira, itd.).

Ključne riječi: Polargraph, Arduino, pisači stroj, crtaći stroj, V-tip crtač, polarni crtač, biangularni crtač, biangularni koordinatni sustav

ABSTRACT

Title: Automatic drawing machine for school boards

For this project a drawing machine was made that uses the biangular coordinate system instead of the cartesian coordinate system. The Arduino board receives commands from the computer and executes them using the stepper motors. The point of this work is to have a drawing machine that can adapt to its drawing surface easily (change it for a small setup, a large setup, a setup that uses smooth surfaces like whiteboards, a setup that uses rough surfaces for gluing paper onto it, etc.).

Key words: Polargraph, Arduino, plotter, V-type plotter, polar plotter, biangular plotter, biangular coordinate system

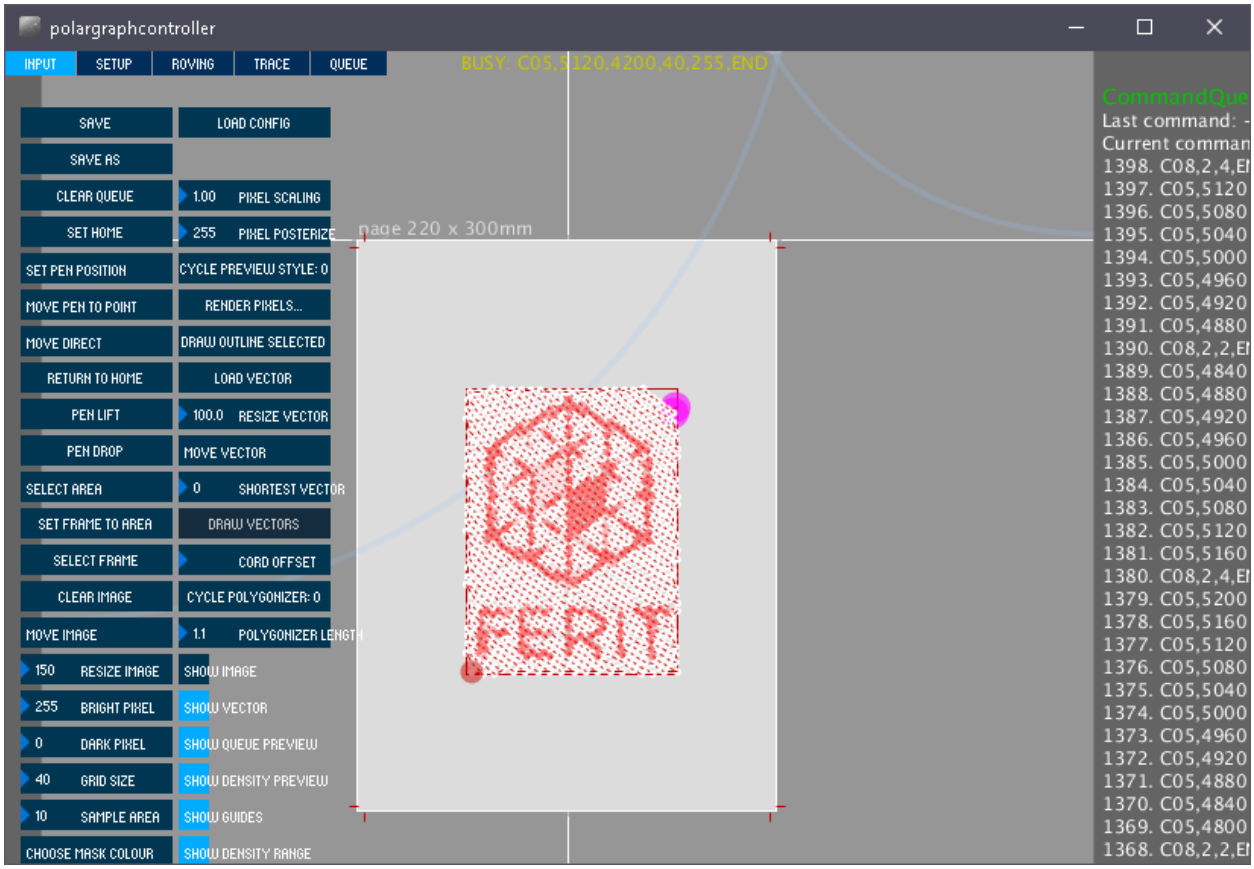
ŽIVOTOPIS

Anto Tufeković je rođen u Sisku 26.01.1998. Početkom je živio u Hrastovcu koji se nalazi u Bjelovarsko-bilogorskoj županiji, ali kasnije se preselio sa obitelji u Tenji u Osječko-baranjskoj županiji. Pohađao je osnovnu školu „Ginko“ u Tenji pa srednju školu „Elektrotehnička i prometna škola Osijek“ te završava sa vrlo dobrim uspjehom. 2016. godine se upisuje u „Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek“ (FERIT).

_____

Anto Tufeković

PRILOG A: Dodatne slike koje su prevelike za dokument



Slika A.1 – Prikaz načina crtanja po pikselu

Slika A.2 – Prikaz crteža norveškog piksela

PRILOG B: Python program za izračun napetosti i rezolucije

```
# Original code by Bill Ola Rasmussen from http://2e5.com/plotter/V/design/
# Originalni kod od Bill Ola Rasmussen sa stranice http://2e5.com/plotter/V/design/
import sys
from PIL import Image, ImageDraw
from math import sqrt, sin, cos, acos, atan2

# setup the constants
version = 1.7
outputFile = "out.png"
width, height = 800, 600
border = 32
# V line end points
v1 = border / 2, border / 2
v2 = width - border / 2 - 1, border / 2


def cross(draw, p, n):
    c = "#000000"
    draw.line((p[0], p[1] - n, p[0], p[1] + n), c)
    draw.line((p[0] - n, p[1], p[0] + n, p[1]), c)


def draw_fixtures(draw):
    # border of calculation pixels
    draw.rectangle([border - 1, border - 1, width - border, height - border],
"#FFFFFF", "#000000")
    # V line end points
    cross(draw, v1, border / 4)
    cross(draw, v2, border / 4)


# Jednadžba 3-1
def line_tensions(a1, a2):
    d = cos(a1) * sin(a2) + sin(a1) * cos(a2)
    return cos(a2) / d, cos(a1) / d


def tension_ok(p):
    # find angles, Jednadžba 3-3
    a1 = atan2(p[1] - v1[1], p[0] - v1[0])
    a2 = atan2(p[1] - v2[1], v2[0] - p[0])
    # strings tension check
    t1, t2 = line_tensions(a1, a2)
    lo, hi = .5, 1.5
    return lo < t1 < hi and lo < t2 < hi


def get_distance(p1, p2):
    return sqrt((p1[0] - p2[0]) ** 2 + (p1[1] - p2[1]) ** 2)


def calculate_point_b(a, b, c):
    alpha = acos((b ** 2 + c ** 2 - a ** 2) / (2 * b * c))
    return b * cos(alpha) + v1[0], b * sin(alpha) + v1[1]
```

```
def resolution_ok(p):
    max_deviation = 1.4
    # law of cosines calculation and nomenclature, Jednadžba 3-2
    c = get_distance(v1, v2)
    b = get_distance(v1, p)
    a = get_distance(v2, p)
    # sanity check
    err = .00000000001
    pc = calculate_point_b(a, b, c)
    assert p[0] - err < pc[0] < p[0] + err
    assert p[1] - err < pc[1] < p[1] + err
    # calculate mapped differences
    db = get_distance(p, calculate_point_b(a, b + 1, c))  # extend left line by 1
unit
    da = get_distance(p, calculate_point_b(a + 1, b, c))  # extend right line by 1
unit
    return db < max_deviation and da < max_deviation  # line pull of 1 unit does not
move x,y by more than max


def calc_pixel(draw, p):
    tension_check = tension_ok(p)
    resolution_check = resolution_ok(p)
    if not tension_check and not resolution_check:
        draw.point(p, "#2620d5")
    if not tension_check and resolution_check:
        draw.point(p, "#4876FF")
    if tension_check and not resolution_check:
        draw.point(p, "#FF7F24")
    # default to background color


def draw_pixels(draw):
    for y in range(border, height - border):
        sys.stdout.write('.')
        sys.stdout.flush()
        for x in range(border, width - border):
            calc_pixel(draw, (x, y))
    sys.stdout.write('\n')


def main():
    image = Image.new("RGB", (width, height), "#D0D0D0")
    draw = ImageDraw.Draw(image)

    draw_fixtures(draw)
    draw_pixels(draw)

    image.save(outputFile, "PNG")


if __name__ == "__main__":
    main()
```