Programiranje mini CNC printera izrađenog od recikliranih elektroničkih komponenti.

Prelac, Dalibor

ISTARSKO VELEUČILIŠTE –

UNIVERSITÀ ISTRIANA DI SCIENZE APPLICATE

Dalibor Prelac

PROGRAMIRANJE MINI CNC PRINTERA IZRAĐENOG OD RECIKLIRANIH ELEKTRONIČKIH KOMPONENTI

Završni rad

Pula, 2024.

ISTARSKO VELEUČILIŠTE –

UNIVERSITÀ ISTRIANA DI SCIENZE APPLICATE

Dalibor Prelac

PROGRAMIRANJE MINI CNC PRINTERA IZRAĐENOG OD RECIKLIRANIH ELEKTRONIČKIH KOMPONENTI

Završni rad

JMBAG: 0233009298, izvanredni student

Studijski smjer: Prijediplomski stručni studij Mehatronike

Predmet: Osnove programiranja

Mentor: Marko Turk, dipl. oec., pred.

Pula, 2024.

SADRŽAJ

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Dalibor Prelac, kandidat za prvostupnika Mehatronike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

U Puli, _____ godine

Student

_____

IZJAVA

o korištenju autorskog djela

Ja, Dalibor Prelac dajem odobrenje Istarskom veleučilištu – Università Istriana di scienze applicate, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom „Programiranje mini CNC printera izrađenog od recikliranih elektroničkih komponenti“ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, _____ godine

Potpis

_____

Zahvala

Na prvom mjestu želim izraziti svoju duboku zahvalnost mentoru, Marku Turku, pred. za nesebičnu pomoć, strpljenje i savjete tijekom pisanja ovog završnog rada. Vaše vodstvo i podrška bili su neprocjenjivi i bez vaše pomoći ovaj rad ne bi bio moguć.

Također bih želio zahvaliti tvrtki Valamar Riviera d.d. i mojim nadređenima iz praonice rublja na iznimnoj prilici i podršci koju su mi pružili tijekom mog studiranja. Njihovo razumijevanje i fleksibilnost omogućili su mi da uspješno balansiram profesionalne i akademske obveze, na čemu sam iznimno zahvalan.

Posebnu zahvalu upućujem svojoj supruzi Kristini i obitelji na njihovoj neizmjernoj podršci, razumijevanju i ljubavi tijekom cijelog mog obrazovnog puta. Bez vaše stalne motivacije, ohrabrenja i žrtve, ovo putovanje ne bi bilo moguće. Vaša vjera u mene bila je moj najveći izvor snage.

Hvala svima od srca.

SAŽETAK

Kreiranje i programiranje mini CNC 2D printera pomoću recikliranih DVD uređaja, Arduino Uno pločice i L293D Motor Shield modula predstavlja inovativan projekt spajanja tehnologije i recikliranja elektronike. Projekt omogućava korisnicima stvaranje vlastitog CNC uređaja za crtanje na ravnoj površini. Ključne komponente uključuju koračne motore, servomotor, Arduino Uno pločicu, L293D Motor Shield modul, žice, vijke, DC konektor za napajanje i kemijsku olovku. Kroz jednostavnu električnu shemu, korisnici mogu precizno kontrolirati pokrete motora, omogućujući im da stvaraju različite crteže i uzorke. Softver poput Inkscape-a koristi se za izradu slika za ispis, dok se Pronterface koristi za upravljanje CNC printerom. Rad naglašava važnost recikliranja elektronike i programiranja te kreativnog pristupa tehnologiji. Osim toga, pruža priliku za učenje i istraživanje osnova numeričke kontrolne obrade.

Ključne riječi: Arduino Uno, CNC printer, Inkscape, koračni motor, Pronterface, servomotor

ABSTRACT

Creating and programming a mini CNC 2D printer using recycled DVD drives, Arduino Uno board and L293D Motor Shield module is an innovative project combining technology and electronics recycling. The project allows users to create their own CNC device for drawing on a flat surface. Key components include stepper motors, servo motor, Arduino Uno board, L293D Motor Shield module, wires, screws, DC power connector and ballpoint pen. Through a simple electrical scheme, users can precisely control the movements of the motor, allowing them to create different drawings and patterns. Software like Inkscape is used to create images for printing, while Pronterface is used to control the CNC printer. This paper emphasizes the importance of electronics recycling, programming of electronics and a creative approach to technology. In addition, it provides an opportunity to learn and explore the fundamentals of numerical control processing.

Keywords: Arduino Uno, CNC printer, Inkscape, Pronterface, servomotor , stepper motor

1. UVOD

Suvremeno doba karakterizira ubrzani razvoj tehnologije i elektronike, što otvara nove mogućnosti u različitim područjima, uključujući proizvodnju, obrazovanje, umjetnost i održivi razvoj. Jedan od inovativnih pristupa u ovim područjima je primjena CNC (Computer Numerical Control) tehnologije, koja omogućava preciznu kontrolu strojeva putem računalnih programa. U ovom radu, fokusirat ćemo se na projekt stvaranja pristupačnog i učinkovitog 2D CNC printera koji koristi reciklirane materijale, te kako takav projekt može pridonijeti održivom razvoju.

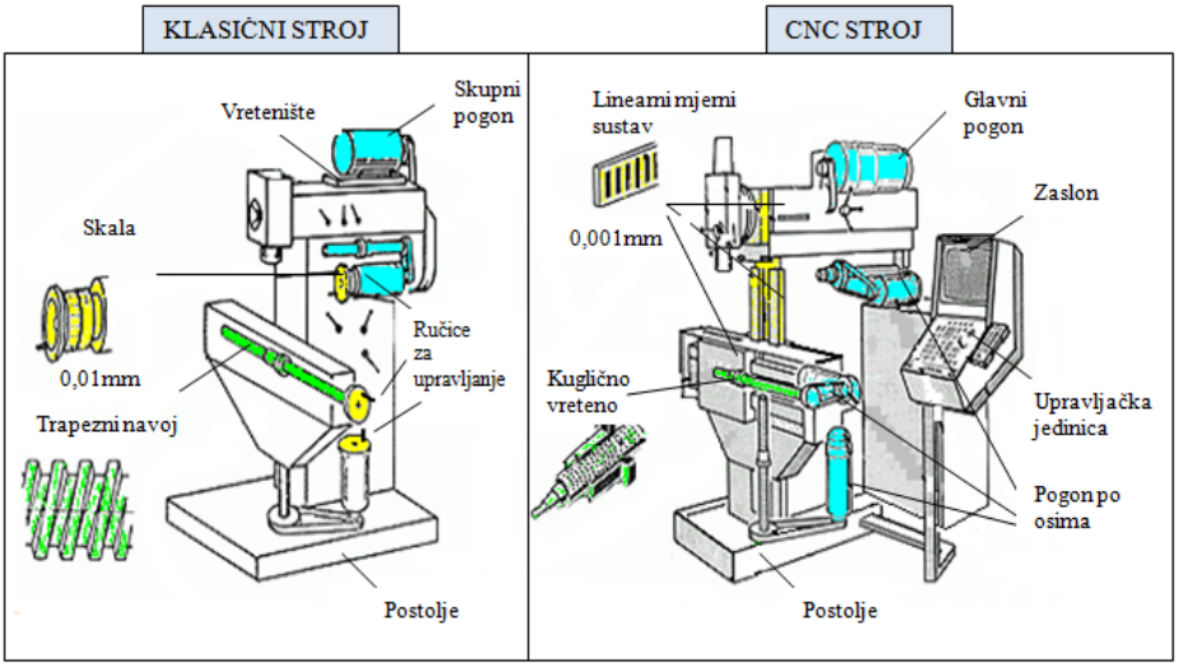
Glavni cilj rada je izraditi funkcionalan 2D CNC printer koristeći dostupne resurse i reciklirane komponente, čime se smanjuje otpad i promiče ekološki prihvatljiv način proizvodnje.

Važno je napomenuti kako se rad sastoji od pet poglavlja. Osim uvodnog poglavlja, drugo poglavlje daje nam uvid u metodologiju izrade, opisujući proces izrade 2D CNC printera, uključujući odabir materijala i komponenti. Treće poglavlje detaljno prikazuje korake izrade i montaže printera. Četvrto poglavlje objašnjava kako programirati CNC printer i provesti testiranja kako bi se osigurala njegova funkcionalnost. Rad završava s petim poglavljem, u kojem se prati rad printera i rješavanje eventualnih neispravnosti tijekom rada.

2. CNC STROJEVI I NJIHOVO PROGRAMIRANJE

Automatizacija alatnih strojeva, kako je poznajemo danas, započela je 1950-ih godina s pojavom NC (Numerical Control) strojeva, odnosno numerički upravljanih strojeva. Ovi strojevi nisu bili računalno upravljani, već su koristili bušene kartice, vrpce ili magnetske vrpce za upravljanje.

CNC (Computer Numerical Control) strojevi koriste mikroprocesore za upravljanje, što omogućuje uređivanje programa izravno na stroju. To su mala elektronička računala koja se mogu programirati za numeričko upravljanje strojem. Numeričko upravljanje podrazumijeva vođenje alatnog stroja pomoću unaprijed definiranog programa.



Slika 1: Razlike između običnog i CNC stroja

Izvor:https://www.oss.unist.hr/Portals/0/adam/Contents/TnPnQrlyXEeYJna9tPx2wQ/Text/PROGRAMIRANJE%20CNC%20STROJEVA%20Sinumerik%20840D.pdf

Klasični i CNC strojevi se značajno razlikuju u načinu rada. Glavna razlika leži u načinu upravljanja. Klasični strojevi se upravljaju ručno ili pomoću ručica, dok se CNC strojevi upravljaju numeričkom kontrolnom jedinicom (CNC) na temelju unaprijed definiranog programa.

To rezultira nizom prednosti CNC strojeva u odnosu na klasične strojeve. CNC strojevi nude veću preciznost i ponovljivost u radu, mogu izrađivati složenije oblike, rade automatski, te su produktivniji i manje zahtijevaju fizički rad operatera. Zahvaljujući navedenim prednostima, CNC strojevi su postali standard u modernim proizvodnim procesima (Pezer, 2022).

Svaki CNC stroj upravlja se pomoću kontrolne jedinice i specifičnog računalnog programa (softvera). Najčešće se koristi kontrolna jedinica SINUMERIK 840D, koja je standard u mnogim srednjim školama. Pored nje, postoje i mnoge druge upravljačke jedinice kao što su: Allen Bradley, Cincinnati, Fanuc, Heidenhain, Fadal, Fagor, Mazak, Mitsubishi, Okuma, Sharnoa, Yasnac i druge. Računalni program WinNC i SINUMERIK 840D, zajedno s upravljačkom jedinicom, čine integrirani sustav za upravljanje CNC strojem (Blažević, 2004).

Programiranje CNC strojeva uključuje pisanje programa prema tehnološkoj dokumentaciji, što se može obaviti ručno ili uz pomoć računala. Ručno programiranje znači ispisivanje programa blok po blok u uređivačkom programu upravljačke jedinice, uz moguće korištenje ciklusa.

S druge strane, programiranje pomoću računala omogućuje automatsko generiranje programa na temelju odabranih parametara. Ovaj način programiranja koristi specijalizirani softver poput CATIA, MASTERCAM, SOLIDCAM, WinCAM i drugih, koji generiraju programski kod. Računalno programiranje smanjuje vrijeme i troškove proizvodnje, ali zahtijeva dobro poznavanje svih glavnih i pomoćnih funkcija zbog mogućnosti editiranja generiranog koda. Stoga je izvrsno poznavanje ručnog programiranja neophodno za uspješno programiranje na ovaj način (Pezer, 2022).

G-code imaj ključnu ulogu u CNC programiranju. Automatizacija CNC obrade ostvaruje se tumačenjem G-code. CNC strojevi ne mogu razumjeti običan jezik, već rade na temelju skupa specifičnih strojnih naredbi. Programeri sastavljaju te naredbe u G-code datoteku kako bi CNC strojevima dali upute za rad.

Mikrokontroler za CNC programiranje unaprijed je programiran sa značenjem svake G-code naredbe. Stoga, kada mikrokontroler pročita određenu naredbu, odmah zna što treba učiniti. Ako neka G-code naredba nije prepoznata od strane CNC mikrokontrolera, neće biti izvršena.

G-code naredbe koriste se u kombinaciji sa svojim pandanima - M-code. G-code kontroliraju kretanje alata CNC stroja, dok M-code upravljaju funkcijama CNC strojeva, poput protoka rashladne tekućine ili promjene alata. G-code i M-code naredbe zajedno čine kompletnu CNC programsku datoteku (3ERP, bez datuma).

Slicing je postupak u kojem se 3D modeli razdvajaju na tanke slojeve koje printer može postupno izraditi. Korisnici imaju mogućnost prilagodbe parametara poput brzine ispisa, debljine sloja i temperature ekstruzije, kako bi postigli optimalnu kvalitetu i brzinu izrade.

Programiranje printera danas uključuje korištenje CAD softvera za kreiranje modela, slicing softvera za generiranje G-code, te često korištenje internetskih platformi za praćenje i upravljanje procesom ispisa. Poznavanje svih ovih aspekata ključno je za uspješno i učinkovito korištenje 3D printera u industriji.

Autori Pabolu i Srinivas su u studenom 2010. godine dizajnirali i implementirali dvodimenzionalni CNC stroj. U članku "Design and Implementation of a Three Dimensional CNC Machine" naglašava se potreba za prilagodljivošću i vrhunskom kvalitetom. Sustav koristi Visual C# na .NET platformi i obuhvaća tri glavne kategorije računalno upravljanih numeričkih kontrolera: Višeprocesorski sustav s ASIC-om; Korisničko sučelje na računalu; Računalo s karticom za kontrolu kretanja. Dizajn ovog sustava je prilagođen korisniku, pruža precizne rezultate i fleksibilan je za upotrebu (Pabolu, Srinivas, 2010).

Sundar Pandian je 2014. godine u radu "A Low-Cost Build-Your-Own CNC Mill Prototype" prikazao niskobudžetni CNC stroj s dvije osi, koristeći Arduino. Korištene komponente je autor dobio kao gotovi set za sastavljanje od američke tvrtke Zen Tool Works (Pandian, 2014).

3. METODOLOGIJA IZRADE MINI CNC PRINTERA

Ovaj projekt spaja koncept recikliranja elektroničkih komponenti sa visokotehnološkom CNC tehnologijom, omogućavajući korisnicima da stvore mini CNC uređaj koristeći dostupne materijale. Kombinirajući reciklirani DVD uređaj s Arduino Uno platformom i L293D Motor Shield modulom, ovaj projekt pruža priliku za istraživanje i eksperimentiranje s osnovama numeričke kontrolne obrade.

Transformiranjem nepotrebne elektronike u funkcionalan mini CNC 2D printer, ovaj projekt nudi mogućnost izrade uređaja sposobnog za crtanje po ravnoj površini. Korištenjem koraka i smjernica ovog projekta, korisnici mogu ovladati osnovama CNC tehnologije dok istovremeno doprinose očuvanju okoliša recikliranjem starih uređaja.

Implementacija ovog projekta zahtijeva precizan rad pri spajanju elektroničkih komponenti, programiranje Arduino mikrokontrolera za upravljanje motorima te korištenje L293D Motor Shield modula za kontrolu i zaštitu motora. Kroz ovaj proces, korisnici će naučiti osnove elektronike, programiranja i mehaničkog dizajna, stvarajući tako koristan i zabavan uređaj koji istovremeno demonstrira spoj tehnologije i kreativnosti.

3.1 PLAN IZRADE

Izrada mini CNC printera započinje pripremom potrebnih dijelova i alata, uključujući Arduino Uno, L293D Motor Shield modul, servomotor, koračne motore iz DVD uređaja, napajanje od 7.5V te osobno računalo. Nakon nabave komponenata, potrebno je pripremiti alate za montažu kao što su bušilica, lemilica, odvijači, vijci i matice. Konstrukcija se izrađuje recikliranjem kućišta, te mehanizmima iz DVD uređaja s koračnim motorima te dizajniranjem i 3D ispisom dijelova, ako je potrebno. Koračni motori se montiraju na X i Y os koordinatnog sustava, dok se servomotor postavlja na Z os za kontrolu podizanja i spuštanja kemijske olovke za ispisivanje. Elektroničko povezivanje uključuje spajanje koračnih motora na L293D Motor Shield modul, servomotora na odgovarajući pin modula, te L293D modula na Arduino Uno. Nakon toga, instalira se Arduino IDE i Pronterface softver za upravljanje koračnim motorima, zatim se podešava Pronterface softver za slanje G-code s računala na Arduino. Na računalo se instalira Inkscape softver i G-code plugin za generiranje G-code iz slika i

tekstova. U Inkscape-u učita se željena slika ili tekst, pretvara se u G-code i sprema se na računalo. Arduino Uno se povezuje s računalom putem USB kabela, pokrene Pronterface softver, učitava G-code i pošalje na Arduino Uno. Testiranje printera na papiru ili drugoj čvrstoj površini, prilagođavanje postavka motora i servomotora za postizanje željene preciznosti. U posljednjem koraku dokumentira se postupak izrade i rezultate testiranja. Ovaj metodološki pristup omogućava uspješnu izradu funkcionalnog i povoljnog mini CNC printera koji je koristan za obrazovne svrhe i praktično učenje o CNC tehnologijama.

3.2 TABLICA POTREBNIH DJELOVA ZA IZRADU PRINTERA

	NAZIV KOMPONENTE	MODEL	KOLIČINA (KOM)
1.	RECIKLIRANI DVD ČITAČ	CD ILI DVD	2
2.	KORAČNI MOTOR		2
3.	ARDUINO UNO	SMD R3	1
4.	MOTOR SHIELD MODUL	L293D	1
5.	SERVOMOTOR	TOWER PRO 9G	1
6.	ŽICE	JUMPER	11
7.	VIJAK	M6 50 mm	8
8.	MATICA	M6	24
9.	KEMIJSKA OLOVKA		1
10.	NAPAJANJE	NTS 2250 EuP 3-12 V DC	1
11.	DC KONEKTOR ZA NAPAJANJE	PCB - 5.5x2.1mm DC-005 – ženski	1

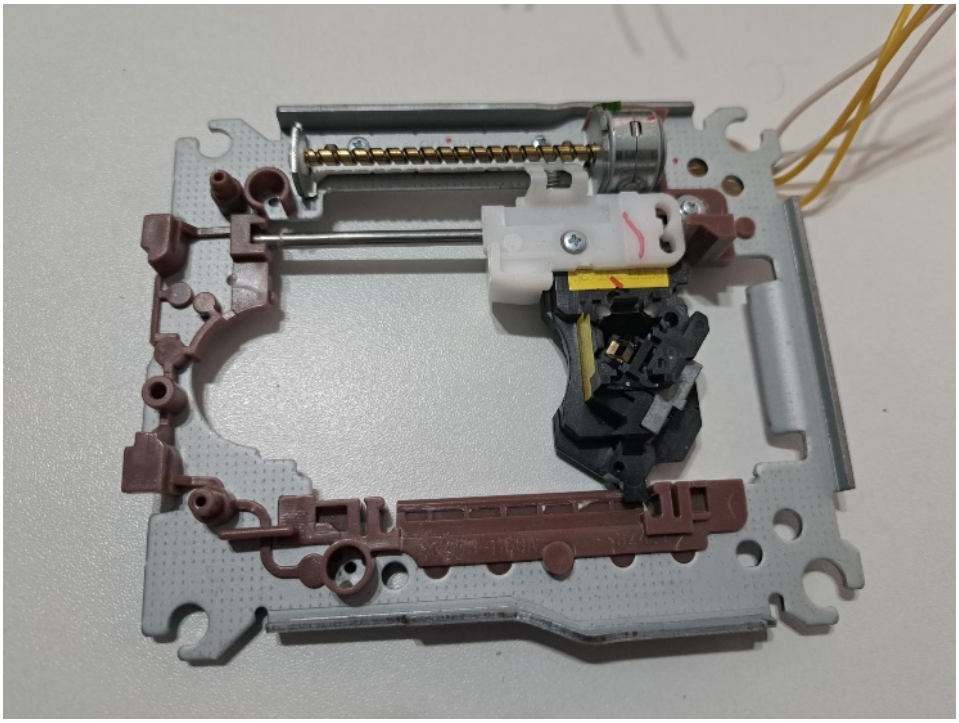
Tablica 1: Popis dijelova i potrebnih količina

Izvor: Autor

3.3 OPIS KOMPONENTI ZA IZRADU PRINTERA

3.3.1 KORAČNI (eng. STEPPER) MOTOR IZ DVD/CD ČITAČA

Koračni (eng. Stepper) motor je vrsta elektromotora, posebno dizajnirana za precizno kontroliranje pozicije i brzine. Ova vrsta motora koristi posebnu tehniku rada koja se temelji na pomacima ili "koracima" koji se izvode kako bi se postigla željena rotacija ili translacija.



Slika 2: Koračni motor i njegovo postolje iz recikliranog DVD/CD čitača

Izvor: Autor

Ključna značajka koračnog motora je da se može precizno pozicionirati bez potrebe za povratnom informacijom o položaju. Sastoji se od elektromagnetskih zavojnica smještenih u statoru i magneta ili zavojnica smještenih na rotoru. Kada se struja primijeni na jednu od zavojnica, dolazi do privlačenja ili odbijanja magneta na rotoru, uzrokujući korak ili pomak u određenom smjeru.

Koračni motori su popularni u sustavima automatskog upravljanja, CNC strojevima, 3D pisačima, robotima i drugim primjenama gdje je potrebna precizna kontrola

pokreta. Mogu se upravljati pomoću posebnih upravljačkih sklopova koji generiraju impulse kako bi pokretali motor kroz definirane korake.

Stare DVD čitače potrebno je rastaviti i iz njih izvući stepper motore zajedno s postoljem koji će se koristiti kao osnova za prijenos kemijske olovke za pisanje i pločice za crtanje.

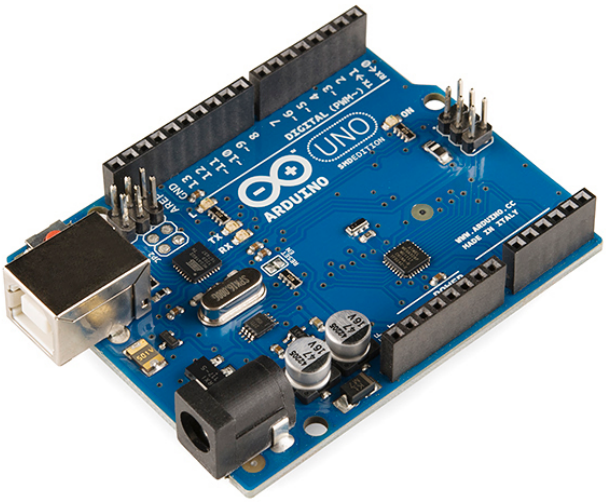
Reciklirana DVD kućišta poslužiti će za postolje CNC uređaja.

Slika 3: Dva komada recikliranih DVD/CD čitača

Izvor: Autor

3.3.2 ARDUINO UNO

Arduino Uno je mikrokontrolerska ploča koja se temelji na ATmega328P čipu, ima 14 digitalnih ulazno-izlaznih pinova, 6 analognih ulaza, 16 MHz keramički rezonator i USB vezu, jednostavan je za korištenje i omogućuje izradu različitih elektroničkih projekata. U ovom projektu arduino pločica funkcionira kao središnji mozak CNC printera, obrađujući naredbe, upravljajući motorima i perifernim uređajima, komunicirajući s računalom te osiguravajući točnost i preciznost kretanja. Bez Arduina ili sličnog mikrokontrolera, koordinacija svih ovih komponenti bila bi vrlo teška i kompliciranа.



Slika 4: Arduino uno, mikrokontrolerska ploča

Izvor: https://www.flickr.com/photos/sparkfun/8406865680/

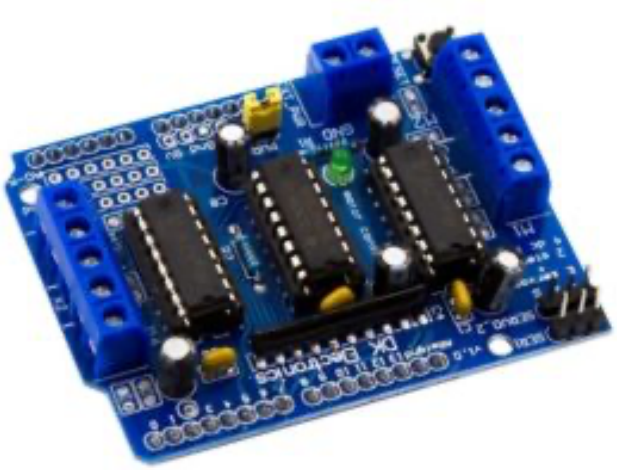
Preuzeto: 15.03.2024.

3.3.3 L293D MOTOR SHIELD

L293D Motor Shield je modul za kontrolu motora, koji se često koristi s mikrokontrolerima poput Arduino. Ovaj modul koristi čip L293D, koji je H-bridge integrirani krug, što znači da omogućava kontrolu smjera i brzine motora. H-bridge je elektronički sklop koji omogućuje reverzibilno upravljanje motorima, što znači da možete kontrolirati rotaciju motora u oba smjera.

L293D Motor Shield često ima nekoliko ulaznih i izlaznih priključaka koji se mogu jednostavno spojiti na mikrokontroler poput Arduina. Ovaj modul pruža praktično sučelje za upravljanje motorima bez potrebe za složenim žičanjem. Također pruža zaštitu od povratnog napona i drugih električnih smetnji koje mogu oštetiti mikrokontroler.

Kroz jednostavno programiranje mikrokontrolera, korisnici mogu kontrolirati brzinu i smjer povezanih motora. Ovaj modul se često koristi u projektima koji zahtijevaju preciznu kontrolu motora, kao što su roboti, mobilni roboti, modeli vozila i slični uređaji.



Slika 5: L293D motor shield modul

Izvor: https://www.engineersgarage.com/arduino-l293d-motor-driver-shield-tutorial/

Preuzeto: 15.03.2024.

3.3.4 SERVOMOTOR

Servomotor Tower Pro SG90 (ili sličan poput 9G) je mali, lagani servomotor koji se često koristi u elektroničkim projektima, modelarstvu, robotici i drugim aplikacijama gdje je potrebna precizna kontrola položaja. Nekoliko ključnih informacija o Tower Pro SG90 servomotoru:

1. Vrsta motora: Tower Pro SG90 je mikro servomotor, što znači da je kompaktan i dizajniran za precizne pokrete.
2. Položajni opseg: Tipično ima položajni opseg od 0 do 180 stupnjeva, što znači da može rotirati u tom rasponu.

3. Radno napajanje: Radi na niskom naponu, obično od 4.8V do 6V, što ga čini prikladnim za većinu mikrokontrolera, uključujući Arduino.
4. Upotreba signala PWM-a: Servomotori poput SG90 koriste PWM (Pulse Width Modulation) signal za kontrolu položaja. Duljina pulsa signala određuje kut rotacije
5. Torzijski moment: Ima ograničen torzijski moment, ali obično je dovoljan za manje zadatke, poput pokretanja ruku robota ili upravljanja kamerom na modelu.
6. Lagani i kompaktni dizajn: SG90 servomotori su lagani, kompaktni i lako se integriraju u različite projekte.
7. Jednostavna kontrola: Kontrola ovog servomotora je relativno jednostavna, često se koristi Arduino biblioteka ili sličan mikrokontroler za generiranje PWM signala i postizanje željenog položaja.

Ovaj servomotor često se koristi za kontrolu mehaničkih dijelova u projektima, poput pokretnih ruku, kotača, kamera ili bilo kojeg drugog mehanizma koji zahtijeva preciznu kontrolu položaja. Zadatak servomotora u ovom projektu je podizanje kemijske olovke tijekom ispisivanja zadanog crteža kako bi se spriječilo iscrtavanje nepotrebnih linija tijekom rada stroja.

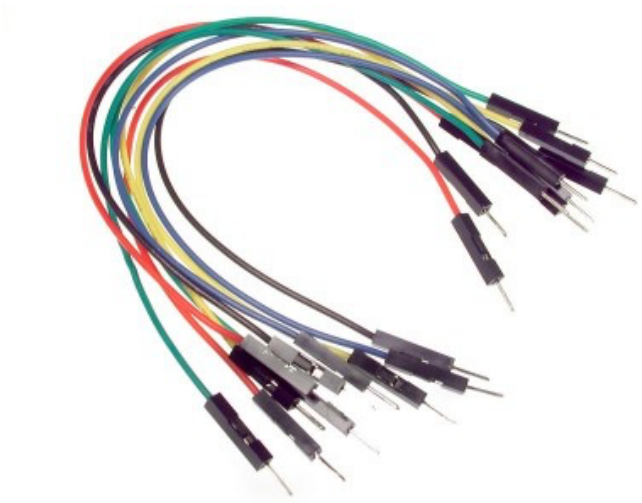
Slika 6: Servomotor Tower pro 9G

Izvor: https://www.amazon.in/Robodo-Electronics-Tower-Micro-Servo/dp/B00MTFFAE0?th=1

Preuzeto: 15.03.2024.

3.3.5 ŽICE

Žice koje se koriste za spajanje Arduino pločica nazivaju se jumper žice. One su kratke, izolirane žice s metalnim vrhovima. Koriste za spajanje različitih komponenata na Arduino pločicu, kao što su LED diode, otpornici, senzori i motori. Obično su dostupne u različitim bojama kako bi se olakšalo praćenje različitih spojeva.



Slika 7: Žice za povezivanje komponenti

Izvor: https://www.exploringarduino.com/parts/jumper-wires/

Preuzeto: 15.03.2024.

3.3.6 VIJCI I MATICE

Vijci i matice potrebni su za montažu koračnih motora i postolja na DVD kućište. Koriste se vijci M6 debljine i dužine 50 mm. Potrebno je imati osam vijka i 24 matice.

Slika 8: Vijak i matica

Izvor: https://pevex.hr/maticni-vijak-s-maticom-m6x50mm-din931-25-1

Preuzeto: 18.03.2024.

3.3.7 KEMIJSKA OLOVKA

Za crtanje po papiru poslužiti će obična kemijska olovka.

Slika 9: Kemijska olovka za crtanje

Izvor: https://www.kult-vk.hr/pisaci-pribor-kemijske-ol-roleri-markeri-tehnolovke-grafitne-olovke/13-kemijska-olovka-fornax-f-070-gumirana-3856000458757.html

Preuzeto: 18.03.2024

3.3.8 NAPAJANJE 7.5 V

GOOBAY 59030 adapter za napajanje je univerzalno napajanje koje može dati izlazni napon od 3 do 12 V DC i maksimalnu struju od 2250 mA. Radi na ulaznom naponu od 100-240V, što ga čini pogodnim za korištenje u većini zemalja. Ovo napajanje je

praktično za različite elektroničke uređaje kao što su računala, igračke, punjači i druge male elektroničke komponente koje zahtijevaju pouzdano napajanje.

Slika 10: Napajanje za mini CNC printer

Izvor: https://www.svijet-medija.hr/art/strujni-adapter-goobay-59030-3-12v-dc-2250ma-100-240v-univerzalni/88767

Preuzeto: 18.03.2024.

3.3.9 DC KONEKTOR ZA NAPAJANJE

Konektor ima dimenzije 5.5x2.1mm, što je standardna veličina za mnoge uređaje, uključujući Arduino pločice, elektroničke projekte, itd. Ovaj konektor se često koristi na PCB-ima (Printed Circuit Boards) kako bi omogućio jednostavno povezivanje i odvajanje napajanja sa samog uređaja. Važno je obratiti pažnju na polaritet konektora. Obično se središnji pin odnosi na pozitivan (+) pol, dok je vanjski prsten negativan (-). DC-005 konektori su široko korišteni u elektronici zbog svoje pouzdanosti i jednostavne upotrebe.

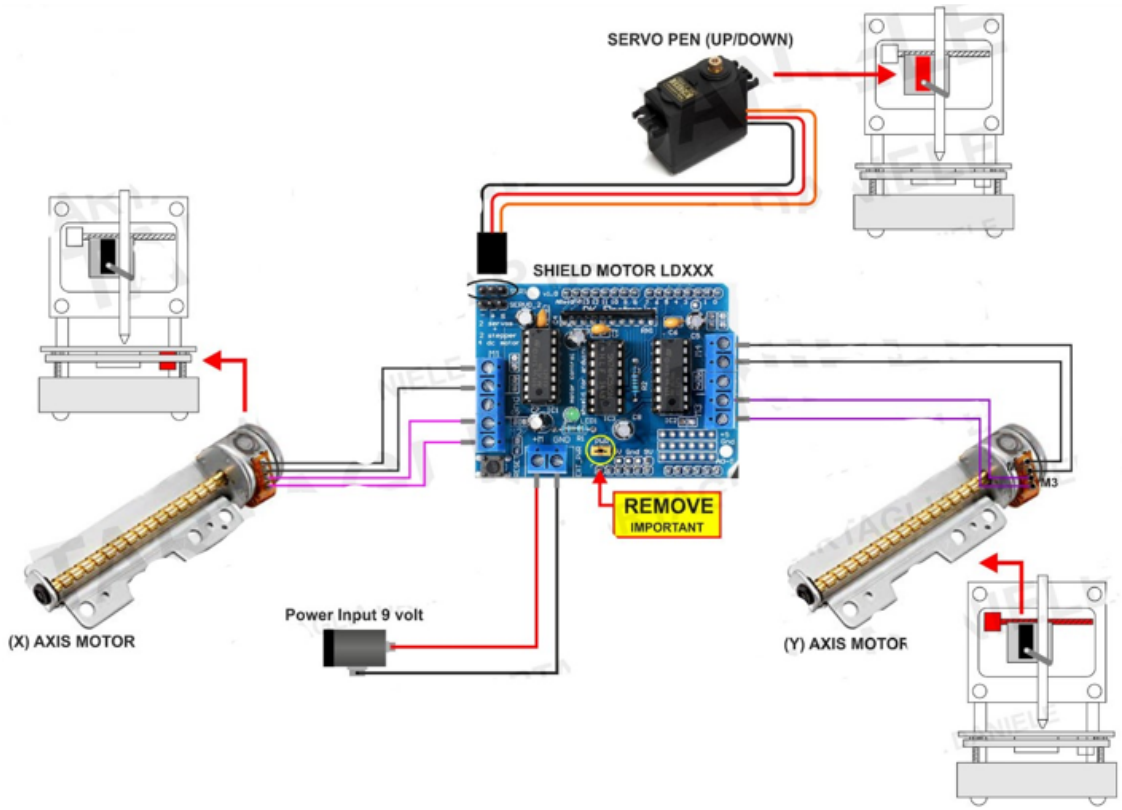
Slika 11: DC konektor za napajanje

Izvor: https://ardubotics.eu/hr/ostalo/1807-dc-zenski-konektor-za-napjanje-pcb-5521mm-dc-005.html

Preuzeto: 18.03.2024.

3.4 ELEKTRIČNA SHEMA SPAJANJA KOMPONENTI

Električni shema je vrlo jednostavna, jedino što treba uzeti u obzir je premosnik koji se mora ukloniti kada napajamo L293D modul preko vanjskog izvora napajanja u ovom slučaju, 7.5 V (volti). Ako se strujni krug napaja s pet volti, motori neće imati dovoljno snage, dok pri višim naponima postoji opasnost od spaljivanja oba koračna motora.



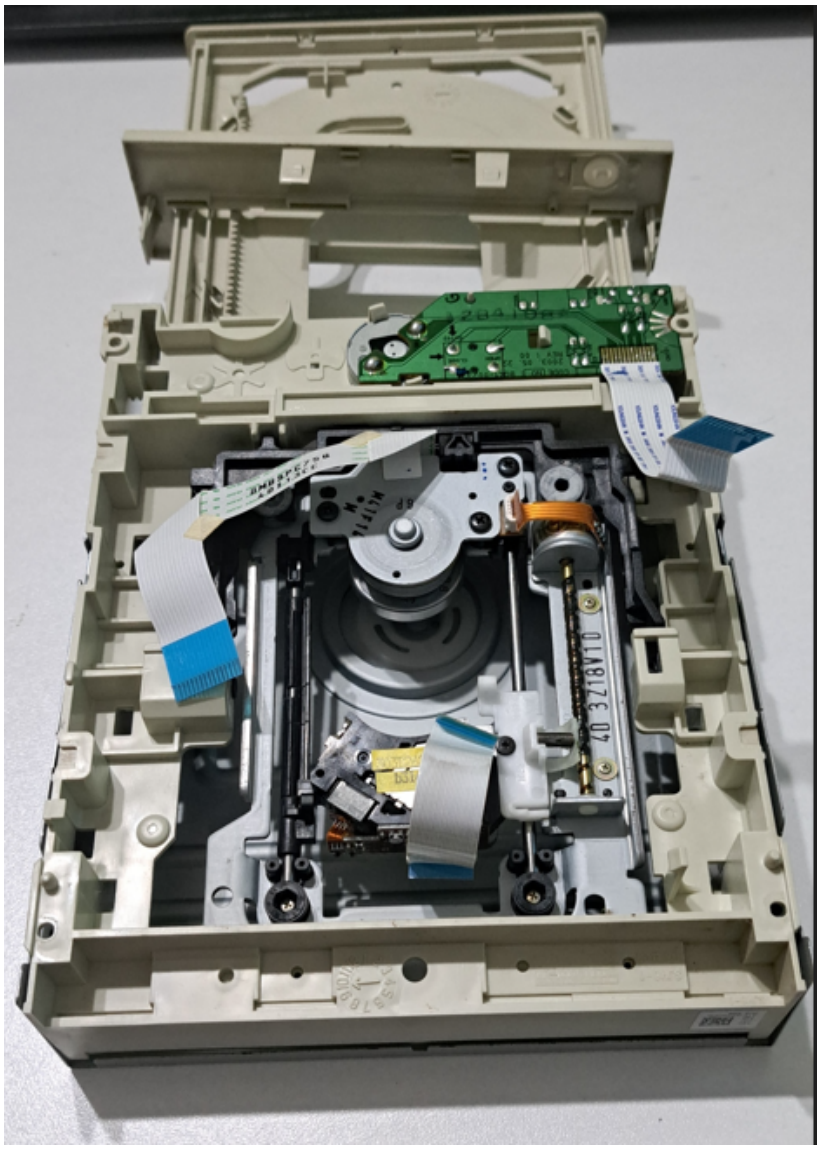
Slika 12: Električna shema spajanja komponenti (autor: Daniele Tartaglia)

Izvor: https://www.labdomotic.com/2019/04/30/youtube-guida-software-cnc-diy/

Preuzeto: 18.03.2024.

3.5 POSTUPAK IZRADE MINI CNC PRINTERA

Na samom početku izrade printera potrebno je rastaviti reciklirane DVD/CD čitače i odvojiti koračne motore zajedno s postoljem od plastične konstrukcije unutar čitača. Rastavljanjem DVD/CD čitača odvajaju se elektroničke komponente od plastičnih. Na taj način omogućuje se recikliranje elektroničkog otpada.

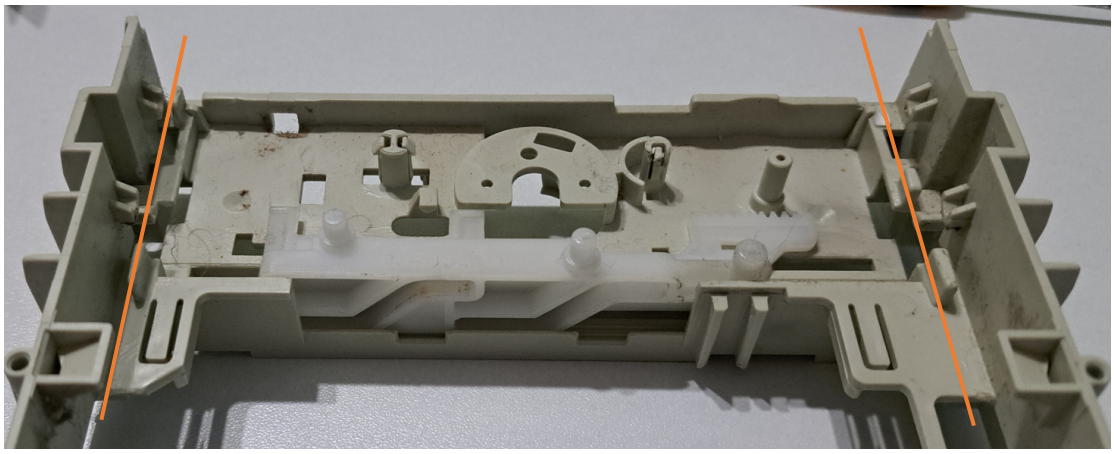


Slika 13: Plastična konstrukcija u unutrašnjosti kućišta DVD/CD čitača

Izvor: Autor

Plastična konstrukcija je sačuvana zbog korištenja u daljnjim koracima izrade. Iz plastičnog mehanizma za izbacivanje DVD/CD diska potrebno je rezati pilom plastiku

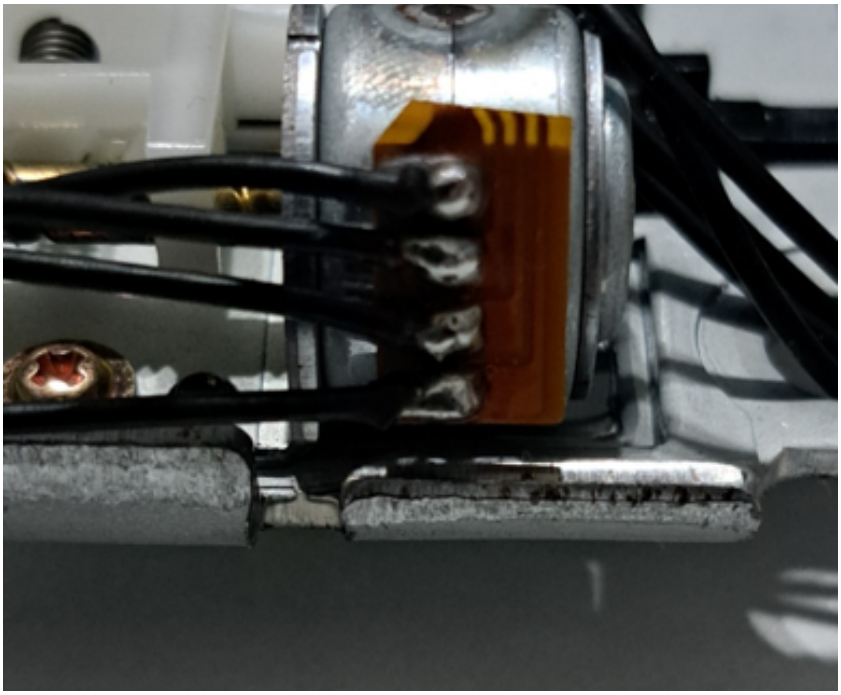
i formirati nosač za kemijsku olovku. Na taj način omogućuje se pomicanje kemijske olovke za pisanje korištenjem servomotora. Na slici 14. narančastim crtama označeno je mjesto gdje je potrebno ispiliti plastiku.



Slika 14: Plastični mehanizam za izbacivanje DVD/CD diska

Izvor: Autor

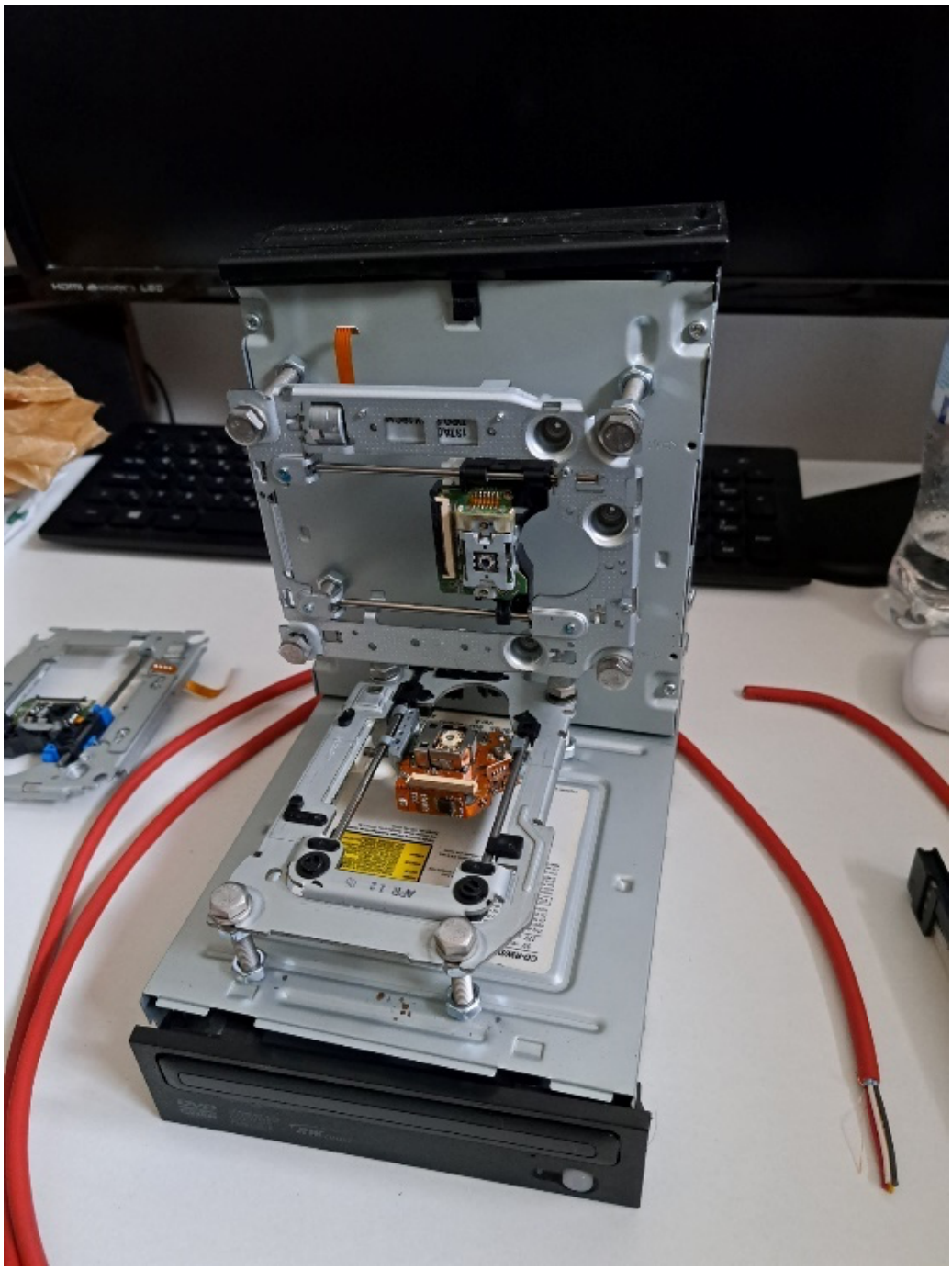
Nakon odvajanja motora i postolja potrebno je na koračni motor zalemiti žice koje služe za napajanje motora.



Slika 15: Zalemljene žice na stepper motor

Izvor: Autor

U sljedećem koraku označavaju se mjesta na DVD kućištu koja je potrebno uz pomoć bušilice izbušiti. Reciklirane DVD/CD kućišta zajedno se učvršćuju i na taj način izrađuje se postolje za koračne motore. Koristeći vijke i matice postavljaju se koračni motori s postoljem na konstrukciju spojenih DVD/CD čitača, vidljivo na slici 16.



Slika 16: Postavljanje koračnih motora s postoljem na konstrukciju

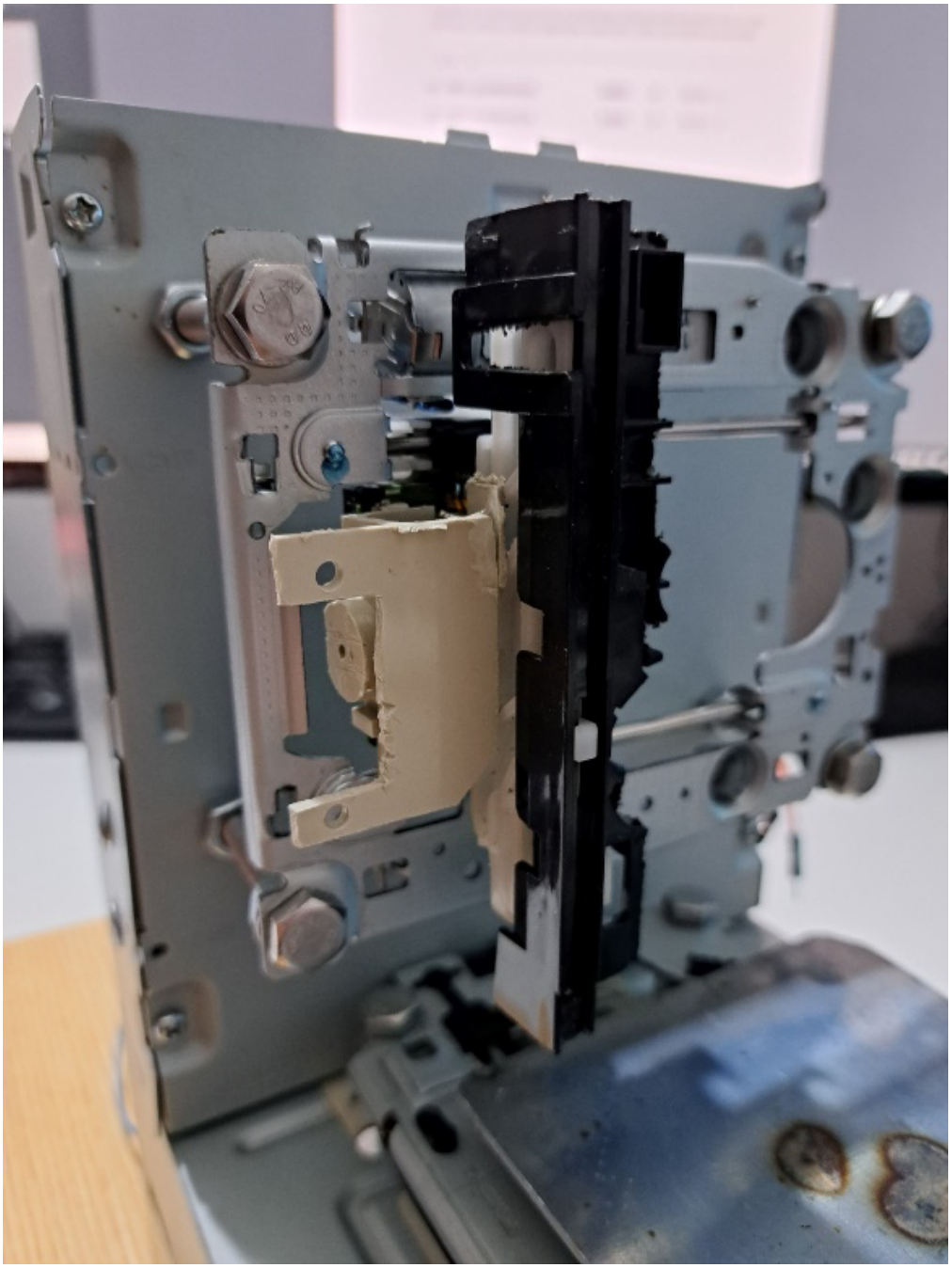
Izvor. Autor

Na vodoravni koračni motor postavlja se pločica za pisanje. U ovom slučaju pločica je od aluminija a moguće je koristiti i različite materijale poput plastike ili drvene pločice. Prednosti aluminijske pločice su mogućnost fiksiranja papira za crtanje korištenjem manjih magneta.

Slika 17: Pločica za držanje papira od aluminija

Izvor. Autor

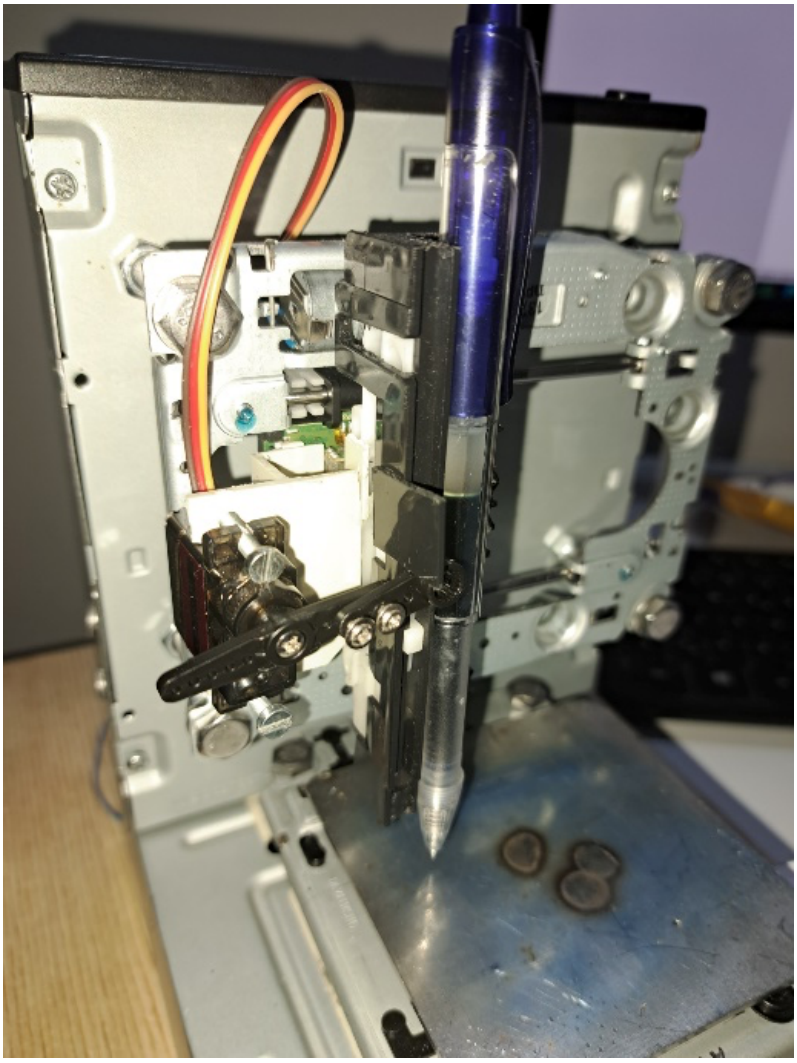
Na okomiti koračni motor postavlja se nosač za kemijsku olovku i servomotor. Nosač je izrađen od plastičnih komponenti sačuvanih prilikom rastavljanja DVD čitača koji su prije služili kao sustav za izbacivanje i ubacivanje DVD/CD-a u kućište čitača. Nosač za kemijsku olovku moguće je učvrstiti za pomični dio nosača koračnog motora uz pomoć manjih vijaka ili ljepilom.



Slika 18: Plastični nosač za kemijsku olovku i servomotor

Izvor. Autor

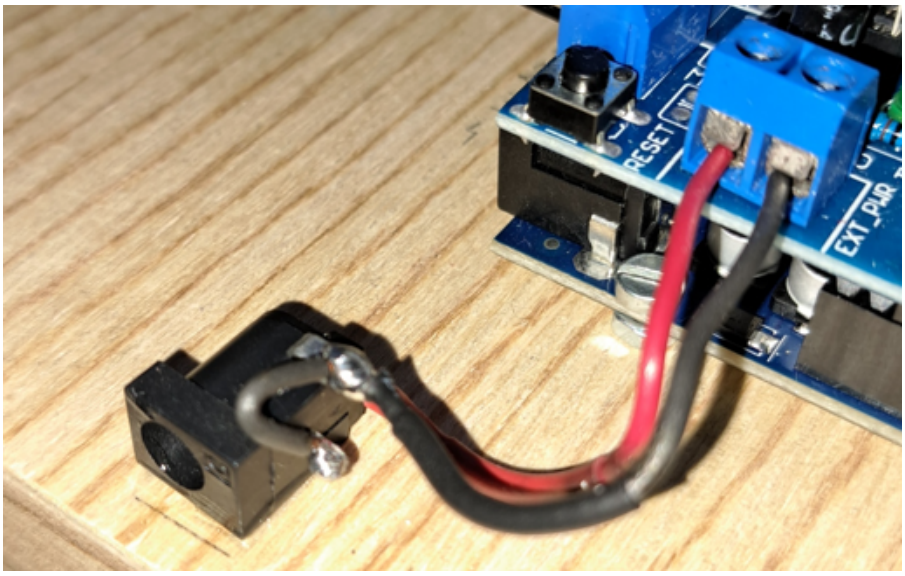
Kemijska olovka se postavlja pomoću ljepila, a za servomotor koristimo dva manja vijka i matice (slika 19).



Slika 19: Pozicioniranje kemijske olovke i servomotora na plastični nosač

Izvor. Autor

DC konektor za napajanje spaja se na modul zbog potrebe napajanja koračnih i servomotora. U slučaju da motori ne obavljaju zadani pokret, postoji mogućnost nedovoljnog napajanja te je potrebno podići napon na 9 volti. Kao kod koračnog motora potrebno je zalemiti žice na DC konektor i omogućiti lakše spajanje.

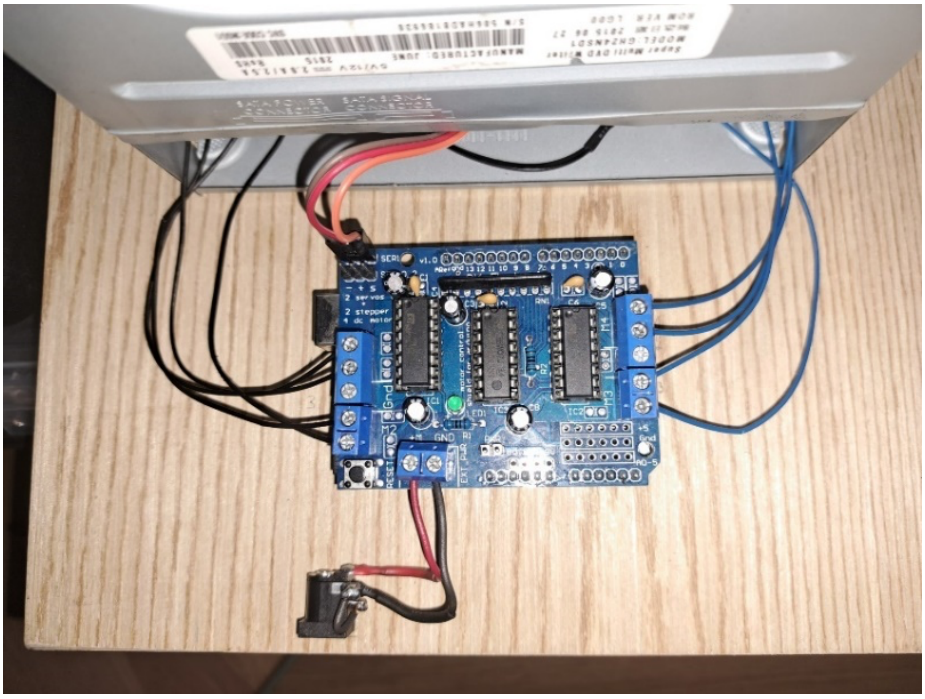


Slika 20: DC konektor povezan na modul

Izvor. Autor

Koristeći električnu shemu, potrebno je spojiti koračne motore na L293D Motor Shield modul. Plave žice predstavljaju koračni motor koji pomiče kemijsku olovku lijevo i desno. Crne žice su za koračni motor koji pomiče pločicu za papir naprijed i nazad.

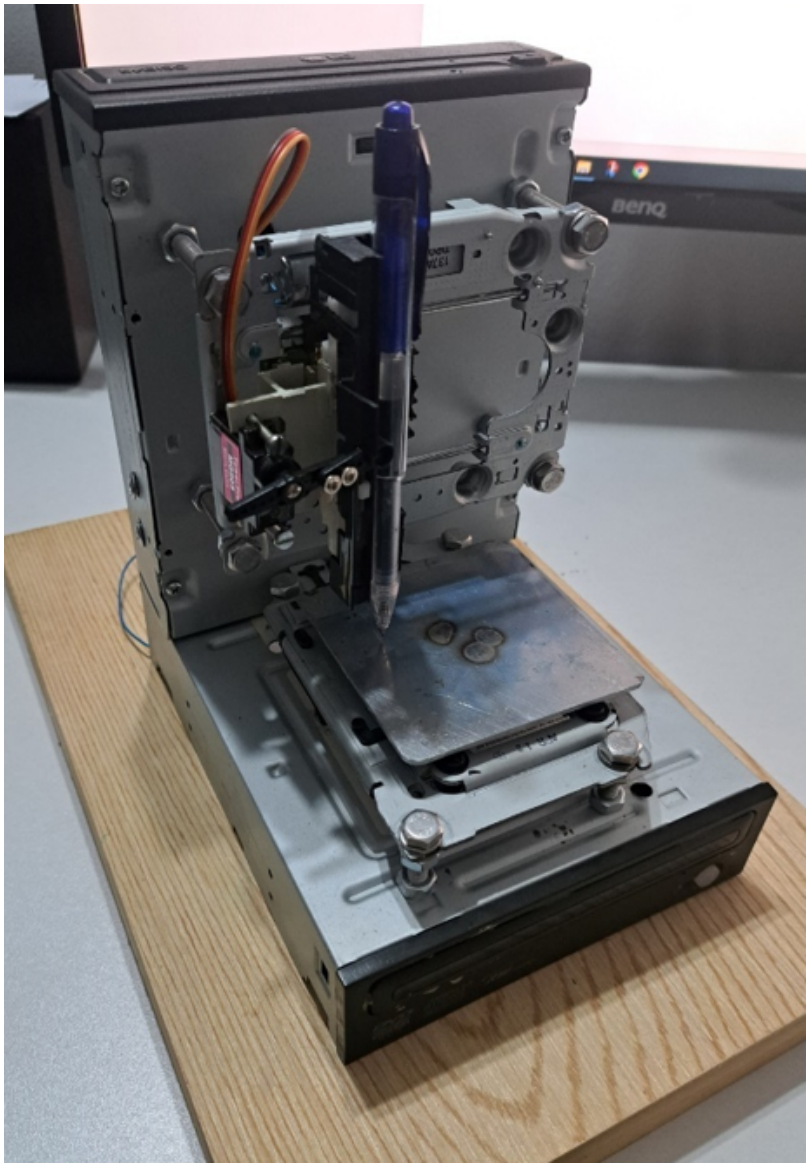
U gornjem lijevom kutu L293D modula spajamo servomotor na za to predviđeno mjesto pomoću pinova na modulu.



Slika 21: Prikaz spojenih električnih žica svih elemenata

Izvor. Autor

CNC printer sada je sastavljen (slika 22) i sve elektroničke komponente su povezane u stroj koji je spreman za povezivanje na osobno računalo.



Slika 22: Sastavljeni mini CNC printer

Izvor. Autor

U posljednjem koraku povezuje se arduino mikrokontroler na stolno računalo ili laptop uz pomoć usb kabla koji dolazi u paketu s arduino pločicom.

Nakon povezivanja arduino pločice s računalom potrebno je učitati kod i na taj način omogućiti kontrolu koračnih i servomotora.

Slika 23: Usb kabel za povezivanje arduino pločice s računalom

Izvor. https://www.amazon.in/Robu-Feet-Cable-Arduino-MEGA/dp/B074FX5M67

Preuzeto: 18.03.2024.

4. PROGRAMSKO RJEŠENJE

4.1 PROGRAMSKI KOD ZA ARDUINO MIKROKONTROLER

Za upravljanje koračnih i servomotora potrebno je na Arduino pločicu učitati programski kod. U daljnjem tekstu naveden je kod i definirane su značajke pojedinih segmenata i redova koda.

Na početku potrebno je uvesti biblioteke koje se koriste u programu. Biblioteka *Servo.h* služi za upravljanje servomotora, dok *AFMotor* upravlja koračnim motorima.

```
#include <Servo.h>
#include <AFMotor.h>

#define LINE_BUFFER_LENGTH 512
```

U ovom slučaju, makronaredba *LINE_BUFFER_LENGTH* koristi se za definiranje veličine linije niza znakova koja se koristi u funkciji *loop()* za pohranjivanje dolaznih serijskih podataka. Definiranjem makronaredbe, programer može jednostavno promijeniti veličinu niza znakova modificiranjem vrijednosti *LINE_BUFFER_LENGTH* na jednom mjestu, umjesto da traži i zamjenjuje sve instance veličine niza u kodu. Ova naredba definira simboličku konstantu koja ima vrijednost 512.

```
char STEP = MICROSTEP ;
```

Deklariranje varijable *STEP* kao tipa char i dodjeljivanje vrijednosti *MICROSTEP*.

```
const int penZUp = 95;
const int penZDown = 83;
```

Definira se položaj servomotora za podizanje i spuštanje olovke.

```
const int penServoPin =10 ;
```

U ovom slučaju *penServopin* označava pin koji je povezan s servomotorom koji kontrolira pokretanje olovke za pisanje. Korištenje konstante olakšava čitanje koda i omogućava lako ažuriranje pinova u slučaju fizičke promjene hardverske konfiguracije.

```
const int stepsPerRevolution = 48;
```

Definiranje konstante *stepsPerRevolution* kao cjelobrojnu (integer) vrijednost i dodjeljuje joj vrijednost 48. Ključna riječ *const* označava da je ova konstanta nepromjenjiva, što znači da se vrijednost *stepsPerRevolution* ne može mijenjati tijekom izvođenja programa.

Ova vrijednost odnosi se na broj koraka po punom krugu (revoluciji) motoričkog mehanizma. Ovakva konstanta često se koristi u programiranju kada se radi s koračnim motorima ili drugim uređajima koji koriste korake za kontrolu pozicije ili kretanje. U ovom slučaju, broj 48 predstavlja broj koraka potreban da se izvrši jedna puna revolucija motora.

```
Servo penServo;
```

Linija koda stvara objekt tipa *Servo* i dodjeljuje mu naziv *penServo*. Nakon izvršavanja ove naredbe, objekt *penServo* će biti spreman za korištenje u programu za kontrolu servomotora povezanog s Arduinom.

```
AF_Stepper myStepperY(stepsPerRevolution,1);
AF_Stepper myStepperX(stepsPerRevolution,2);
```

Ove dvije linije programskog koda inicijaliziraju koračne motore za kontrolu pomoću Adafruit Motor Shield-a. Koristeći odgovarajuće pinove i definiran broj koraka po revoluciji. Nakon inicijalizacije, ovi objekti mogu se koristiti za kontrolu pokreta koračnog motora u programu, na primjer, za kretanje u određenom smjeru ili do određene pozicije.

```
struct point {
  float x;
  float y;
  float z;
};
```

Naredba definira strukturu podataka nazvanu *point*. Struktura je oblik podataka koja omogućuje grupiranje različitih varijabli različitih tipova podataka pod jedan naziv.

U ovom slučaju, struktura *point* ima tri člana:

- float x;: član strukture tipa float koji predstavlja koordinatu x točke.
- float y;: član strukture tipa float koji predstavlja koordinatu y točke.
- float z;: član strukture tipa float koji predstavlja koordinatu z točke.

```
struct point actuatorPos;
```

Stvaranje varijable koja predstavlja trenutni položaj pisaćeg alata.

```
float StepInc = 1;
int StepDelay = 0;
int LineDelay =0;
int penDelay = 50;
```

Definiranje varijable *StepInc* kao decimalni broj (float) i dodjeljuje joj se početna vrijednost 1. Ova varijabla predstavlja inkrement za korake, za kontrolu pomicanja uređaja ili pozicioniranje u procesu. Varijable *StepDelay* definirana je kao cjelobrojni broj (integer) i dodjeljuje joj se početna vrijednost 0. Ova varijabla predstavlja kašnjenje između koraka u procesu kretanja. Definiranje varijable *LineDelay* kao cjelobrojni broj (integer) i dodjeljuje joj se početna vrijednost 0. Ova varijabla predstavlja kašnjenje između linija u procesu. Definiranje varijable *penDelay* kao cjelobrojni broj (integer) i dodjeljuje joj se početna vrijednost 50. Ova varijabla predstavlja kašnjenje ili trajanje vremena za koje se olovka zadržava u određenom položaju prije nego što se izvrši neka druga radnja.

```
float StepsPerMillimeterX = 100.0;
float StepsPerMillimeterY = 100.0;
```

Definiraju se dvije varijable, *StepsPerMillimeterX* i *StepsPerMillimeterY*, koje su obje tipa float (tj. decimalni brojevi s pomičnim zarezom).

Varijabla *StepsPerMillimeterX* postavljena je na vrijednost 100.0, ova varijabla predstavlja broj koraka motora potrebnih za prijeđeni put od 1 milimetra po osi X. Varijabla *StepsPerMillimeterY* također je postavljena na 100.0, ona predstavlja broj koraka motora potrebnih za prijeđeni put od 1 milimetra po osi Y.

```
float Xmin = 0;
float Xmax = 40;
float Ymin = 0;
float Ymax = 40;
float Zmin = 0;
float Zmax = 1;

float Xpos = Xmin;
float Ypos = Ymin;
float Zpos = Zmax;
```

Definiranje *float* varijabli koje se koriste za postavljanje granica i trenutnih pozicija u trodimenzionalnom prostoru rada mini CNC printera.

Varijable Xmin, Xmax, Ymin, Ymax, Zmin i Zmax predstavljaju granice prostora u smjerovima X, Y i Z. U ovom slučaju, Xmin i Ymin su postavljeni na 0, što znači da je to početna točka u koordinatnom sustavu. Xmax, Ymax i Zmax postavljeni su na 40, što znači da je maksimalna veličina prostora 40 po svakoj od tih dimenzija.

Varijable Xpos, Ypos i Zpos predstavljaju trenutne pozicije u trodimenzionalnom prostoru. U ovom slučaju, Xpos i Ypos postavljeni su na Xmin i Ymin, što znači da je trenutna pozicija na početnoj točki prostora (0, 0). Zpos postavljen je na Zmax, što sugerira da je trenutna pozicija u smjeru Z na maksimalnoj vrijednosti.

Ove varijable se često koriste u sustavima za upravljanje CNC strojevima ili drugim uređajima koji se kreću u trodimenzionalnom prostoru kako bi se precizno kontrolirale pozicije i kretanja.

```
boolean verbose = false;
```

Linija koda definira varijablu *verbose* koja je tipa *boolean*.

Varijabla *verbose* se koristi u programiranju kao način da se omogući ili onemogući detaljno izvješćivanje ili ispis informacija tijekom izvođenja programa. Kada je *verbose*

postavljen na *true*, program će izdavati detaljnije izlazne poruke ili informacije. Kada je postavljen na *false*, program će obično izdavati samo osnovne ili nužne informacije.

```
void setup() {

  Serial.begin( 9600 );

  penServo.attach(penServoPin);
  penServo.write(penZUp);
  delay(100);

  myStepperX.setSpeed(600);

  myStepperY.setSpeed(600);
```

Unutar funkcije *void setup* postavlja se brzina kretanja za dva koračna motora, *myStepperX* i *myStepperY*, na 600 koraka u minuti.

Postavljanje brzine kretanja za ove motore pomoću *setSpeed* funkcije omogućuje kontrolu koliko brzo će se motori kretati dok obavljaju određene zadatke ili pokrete. U ovom slučaju, brzina je postavljena na 600 koraka u minuti, ali se može prilagoditi prema potrebama aplikacije ili zahtjevima korisnika.

Važno je prilagoditi brzinu kretanja motora ovisno o zahtjevima aplikacije kako bi se postigla optimalna izvedba i preciznost pokreta. Brzina kretanja može utjecati na kvalitetu rezultata i točnost pozicioniranja, pa je važno odabrati odgovarajuću brzinu ovisno o specifičnim zahtjevima projekta.

```
Serial.println("Mini CNC Plotter alive and kicking!");
  Serial.print("X range is from ");
  Serial.print(Xmin);
  Serial.print(" to ");
  Serial.print(Xmax);
  Serial.println(" mm.");
  Serial.print("Y range is from ");
  Serial.print(Ymin);
  Serial.print(" to ");
  Serial.print(Ymax);
  Serial.println(" mm.");
}
```

Funkcije *Serial.println()* i *Serial.print()* koriste se za ispisivanje teksta i vrijednosti varijabli putem serijske veze. Kada se sve ove linije izvrše zajedno, rezultat će biti ispisivanje informacija o rasponima za osi X i Y na serijskoj konzoli.

```
void loop()
{

  delay(100);
  char line[ LINE_BUFFER_LENGTH ];
  char c;
  int lineIndex;
  bool lineIsComment, lineSemiColon;

  lineIndex = 0;
  lineSemiColon = false;
  lineIsComment = false;
```

Unutar *loop()* funkcije definiraju se nekoliko varijabli koje će se koristiti. *lineIndex:* varijabla koja će pratiti trenutni indeks u nizu *line*. Koristi se za pohranu znakova koje program prima putem serijske veze.

lineSemiColon: je varijabla tipa bool (boolean) koja označava je li trenutna linija u obradi sadrži znak ";". Ovo je korisno za različite svrhe obrade linija teksta, za razdvajanje naredbi ili druge svrhe.

lineIsComment: je varijabla tipa bool koja označava je li trenutna linija u obradi komentar. U programiranju, komentari su dijelovi koda koji su ignorirani prilikom izvođenja, a koriste se za dokumentiranje ili isključivanje dijelova koda bez brisanja. Ova varijabla omogućuje programu da prepozna komentare i ponaša se prema njima u skladu s tim.

Varijable će se koristiti za obradu linija teksta koje dolaze putem serijske veze ili iz nekog drugog izvora, omogućujući programu da analizira te linije i reagira na odgovarajući način, ovisno o sadržaju linije.

```
while (1) {

    while ( Serial.available()>0 ) {
      c = Serial.read();
      if (( c == '\n') || (c == '\r') ) {
        if ( lineIndex > 0 ) {
          line[ lineIndex ] = '\0';
```

```
            if (verbose) {
              Serial.print( "Received : ");
              Serial.println( line );
            }
            processIncomingLine( line, lineIndex );
            lineIndex = 0;
          }
          else {

          }
          lineIsComment = false;
          lineSemiColon = false;
          Serial.println("ok");
        }
        else {
          if ( (lineIsComment) || (lineSemiColon) ) {
            if ( c == ')' )  lineIsComment = false;
          }
          else {
            if ( c <= ' ' ) {
            }
            else if ( c == '/' ) {
            }
            else if ( c == '(' ) {
              lineIsComment = true;
            }
            else if ( c == ';' ) {
              lineSemiColon = true;
            }
            else if ( lineIndex >= LINE_BUFFER_LENGTH-1 ) {
              Serial.println( "ERROR - lineBuffer overflow" );
              lineIsComment = false;
              lineSemiColon = false;
            }
            else if ( c >= 'a' && c <= 'z' ) {
              line[ lineIndex++ ] = c-'a'+'A';
            }
            else {
              line[ lineIndex++ ] = c;
            }
          }
        }
      }
    }
  }
```

Ovaj dio koda predstavlja beskonačnu petlju (*while (1)* ili *while(true)*) koja se stalno izvršava. U ovoj petlji obrađuje se serijski ulaz (komunikacija putem serijske veze) i obrađuje se svaka linija teksta koja se šalje putem serijske veze.

Unutar vanjske petlje *while(1)* provjerava se dostupnost podataka na serijskom ulazu. Ovo se postiže pozivom funkcije *Serial.available()*.Ako ima dostupnih podataka na serijskom ulazu (Serial.available() > 0), podaci se čitaju jedan po jedan pomoću

funkcije *Serial.read()*.Ako se pročita znak novog reda (c == '\n') ili povratni znak (c == '\r'), to znači da je završena jedna linija teksta.

Ako je *lineIndex* (indeks u nizu line) veći od nule, to znači da je linija kompletna i spremna za obradu. U suprotnom, preskače se blok jer linija ostaje prazna ili je komentar. Ako je *verbose* postavljen na *true*, ispisuje se primljena linija teksta.

Nakon toga, linija teksta šalje se funkciji *processIncomingLine()* na obradu. *lineIsComment* se postavlja na *false* jer je završena linija i više nije komentar. *lineSemiColon* se također postavlja na *false*.

Na kraju, šalje se odgovor "ok" putem serijske veze kako bi se potvrdilo da je linija primljena i obrađena. Ostatak koda obrađuje različite slučajeve koji se mogu pojaviti tijekom obrade linija teksta, kao što su komentari, znakovi za kraj linije, znakovi za brisanje, itd. Ovisno o njihovom sadržaju, linija teksta se priprema za obradu ili se ignorira.

```
void processIncomingLine( char* line, int charNB ) {
  int currentIndex = 0;
  char buffer[ 64 ];
  struct point newPos;

  newPos.x = 0.0;
  newPos.y = 0.0;
```

Ovaj dio koda definira funkciju *processIncomingLine*, koja se koristi za obradu linija teksta koje su primljene putem serijske veze.

Dio koda priprema funkciju *processIncomingLine* za obradu linije teksta, a varijable koje su definirane unutar funkcije koriste se za privremeno pohranjivanje podataka tijekom obrade linije. Nakon što funkcija *processIncomingLine* bude završena, pozivajući kod može koristiti strukturu *newPos* za daljnju obradu pozicija ili bilo koje druge potrebne podatke.

```
while( currentIndex < charNB ) {
    switch ( line[ currentIndex++ ] ) {
    case 'U':
      penUp();
      break;
    case 'D':
      penDown();
      break;
    case 'G':
      buffer[0] = line[ currentIndex++ ];
```

```
        buffer[1] = '\0';

        switch ( atoi( buffer ) ){
        case 0:
        case 1:
          ,
          char* indexX = strchr( line+currentIndex, 'X' );
          char* indexY = strchr( line+currentIndex, 'Y' );
          if ( indexY <= 0 ) {
            newPos.x = atof( indexX + 1);
            newPos.y = actuatorPos.y;
          }
          else if ( indexX <= 0 ) {
            newPos.y = atof( indexY + 1);
            newPos.x = actuatorPos.x;
          }
          else {
            newPos.y = atof( indexY + 1);
            indexY = '\0';
            newPos.x = atof( indexX + 1);
          }
          drawLine(newPos.x, newPos.y );
          actuatorPos.x = newPos.x;
          actuatorPos.y = newPos.y;
          break;
        }
        break;
      case 'M':
        buffer[0] = line[ currentIndex++ ];
        buffer[1] = line[ currentIndex++ ];
        buffer[2] = line[ currentIndex++ ];
        buffer[3] = '\0';
        switch ( atoi( buffer ) ){
        case 300:
          {
            char* indexS = strchr( line+currentIndex, 'S' );
            float Spos = atof( indexS + 1);

            if (Spos == 30) {
              penDown();
            }
            if (Spos == 50) {
              penUp();
            }
            break;
          }
        case 114:
          Serial.print( "Absolute position : X = " );
          Serial.print( actuatorPos.x );
          Serial.print( "  -  Y = " );
          Serial.println( actuatorPos.y );
          break;
        default:
          Serial.print( "Command not recognized : M");
          Serial.println( buffer );
        }
      }
    }
  }
```

Ovaj dio koda predstavlja petlju koja obrađuje različite naredbe koje se mogu pojaviti u liniji G-code (G-komande) i izvršava odgovarajuće akcije ovisno o naredbi. Slijedi objašnjenje naredbi u ovom djelu koda:

- *while (currentIndex < charNB)* : petlja prolazi kroz svaki znak u liniji G-code, sve dok se ne dođe do kraja linije.
- *switch (line[currentIndex++])* : obrada naredbe koja se nalazi na trenutnom indeksu *currentIndex* u liniji G-code. Nakon što se izvrši, *currentIndex* se povećava za 1 kako bi se prešlo na sljedeći znak u liniji.
- *case 'U'* : ako je naredba 'U', to znači da se olovka treba podići. Poziva se funkcija *penUp().*
- *case 'D'* : ako je naredba 'D', to znači da se olovka treba spustiti. Poziva se funkcija *penDown().*
- *case 'G'* : ako je naredba 'G', provjerava se sljedeći znak u liniji G-code.
- *buffer[0] = line[currentIndex++];* : Čita se sljedeći znak koji bi trebao predstavljati dio G-naredbe.
- *buffer[1] = '\0';* : postavlja se nulti znak u bufferu kako bi se formirala null-terminirana niska.
- *switch (atoi(buffer))* : pretvara se niska u integer kako bi se izvršila odgovarajuća akcija na temelju vrijednosti G-naredbe.
- *case 0:* i *case 1:* ukoliko je G-naredba 0 ili 1, to obično označava kretanje alatnog stroja. Dalje se provjerava ima li u liniji G-code informacija o novoj poziciji (X i Y koordinate), a zatim se poziva funkcija *drawLine()* za crtanje linije i ažurira pozicija aktuatora (*actuatorPos*).
- *case 'M'*: ako je naredba 'M', provjerava se sljedeći znak u liniji G-code.
- *buffer[0], buffer[1] i buffer[2]* se koriste za čitanje sljedećih tri znaka koji bi trebali predstavljati dio M-naredbe.
- *buffer[3] = '\0';* : postavlja se nulti znak u bufferu kako bi se formirala null-terminirana niska.
- *switch (atoi(buffer)) {* : pretvara se niska u integer kako bi se izvršila odgovarajuća akcija na temelju vrijednosti M- naredbe.
- *case 300* : ako je M- naredbe 300, tada se provjerava sljedeći znak u liniji G-code.

- *char* indexS = strchr(line+currentIndex, 'S');* pronalazi se indeks znaka 'S' u liniji G-code.
- *float Spos = atof(indexS + 1);:* izdvaja se vrijednost parametra 'S' iz linije G-code.
- Ako je *Spos* jednako 30, to znači da je olovka spuštena i poziva se funkcija *penDown()*. Ako je *Spos* jednako 50, to znači da je olovka podignuta i poziva se funkcija *penUp().*
- *case 114*: ako je M- naredbe 114, tada se ispisuje trenutna pozicija alatnog stroja putem serijske veze.
- U suprotnom, ispisuje se poruka da naredba nije prepoznata.

```
void drawLine(float x1, float y1) {

  if (verbose)
  {
    Serial.print("fx1, fy1: ");
    Serial.print(x1);
    Serial.print(",");
    Serial.print(y1);
    Serial.println("");
  }
```

Funkcija *drawLine* ispisuje koordinate početka linije (x1, y1) na serijskoj konzoli ako je varijabla *verbose* postavljena na true.

```
if (x1 >= Xmax) {
    x1 = Xmax;
  }
  if (x1 <= Xmin) {
    x1 = Xmin;
  }
  if (y1 >= Ymax) {
    y1 = Ymax;
  }
  if (y1 <= Ymin) {
    y1 = Ymin;
  }
```

Funkcija *if* u kodu provjerava da li su koordinate početka linije (x1, y1) unutar granica definiranih varijablama *Xmin, Xmax, Ymin* i *Ymax*.

```
if (verbose)
  {
    Serial.print("Xpos, Ypos: ");
    Serial.print(Xpos);
    Serial.print(",");
    Serial.print(Ypos);
    Serial.println("");
  }

  if (verbose)
  {
    Serial.print("x1, y1: ");
    Serial.print(x1);
    Serial.print(",");
    Serial.print(y1);
    Serial.println("");
  }
```

U kodu se provjerava da li je varijabla *verbose* postavljena na true. Ako jest, ispisuje koordinate Xpos i Ypos, te koordinate x1 i y1 na serijskoj konzoli. Ovo je korisno za praćenje vrijednosti varijabli tijekom izvođenja programa, omogućavajući detaljan uvid u njihove vrijednosti ako je *verbose* postavljen na true.

```
x1 = (int)(x1*StepsPerMillimeterX);
  y1 = (int)(y1*StepsPerMillimeterY);
  float x0 = Xpos;
  float y0 = Ypos;


  long dx = abs(x1-x0);
  long dy = abs(y1-y0);
  int sx = x0<x1 ? StepInc : -StepInc;
  int sy = y0<y1 ? StepInc : -StepInc;

  long i;
  long over = 0;
```

Programski kod preračunava koordinate x1 i y1 iz mjernih jedinica u korake motora, koristeći faktore *StepsPerMillimeterX* i *StepsPerMillimeterY*. Nakon toga, definira se početna pozicija x0 i y0. Zatim se izračunava promjena koordinata dx i dy, te se postavljaju smjerovi sx i sy koristeći inkrement *StepInc*. Na kraju, definira se varijabla *over* za eventualno prekoračenje koraka.

```
if (dx > dy) {
    for (i=0; i<dx; ++i) {
      myStepperX.onestep(sx,STEP);
      over+=dy;
      if (over>=dx) {
        over-=dx;
        myStepperY.onestep(sy,STEP);
      }
    delay(StepDelay);
    }
  }
  else {
    for (i=0; i<dy; ++i) {
      myStepperY.onestep(sy,STEP);
      over+=dx;
      if (over>=dy) {
        over-=dy;
        myStepperX.onestep(sx,STEP);
      }
      delay(StepDelay);
    }
  }
```

Kod provjerava da li je promjena po koordinati x (dx) veća od promjene po koordinati y (dy). Ako je tako, pokreće se petlja koja se izvršava dx puta. U svakoj iteraciji, motori se pomjeraju u smjeru x (koristeći varijablu sx) i broji se prekoračenje koraka prema y koordinati. Ako je prekoračenje koraka veće ili jednako promjeni x, pokreće se motor u smjeru y. Ako je promjena po y veća, pokreće se petlja koja se izvršava dy puta. U svakoj iteraciji, motori se pomjeraju u smjeru y (koristeći varijablu sy) i broji se prekoračenje koraka prema x koordinati. Ako je prekoračenje koraka veće ili jednako promjeni y, pokreće se motor u smjeru x. Nakon svakog koraka, program čeka određeno vrijeme prije nego što nastavi s izvršavanjem sljedećeg koraka.

```
if (verbose)
  {
    Serial.print("dx, dy:");
    Serial.print(dx);
    Serial.print(",");
    Serial.print(dy);
    Serial.println("");
  }

  if (verbose)
  {
    Serial.print("Going to (");
    Serial.print(x0);
    Serial.print(",");
    Serial.print(y0);
    Serial.println(")");
  }
```

Ovaj dio koda ispisuje informacije o promjeni koordinata dx i dy, koje predstavljaju razlike između početnih (x0, y0) i ciljnih (x1, y1) koordinata. Također, ispisuje se poruka koja navodi koordinate početne pozicije (x0, y0), ako je varijabla *verbose* postavljena na *true*.

```
delay(LineDelay);

  Xpos = x1;
  Ypos = y1;
```

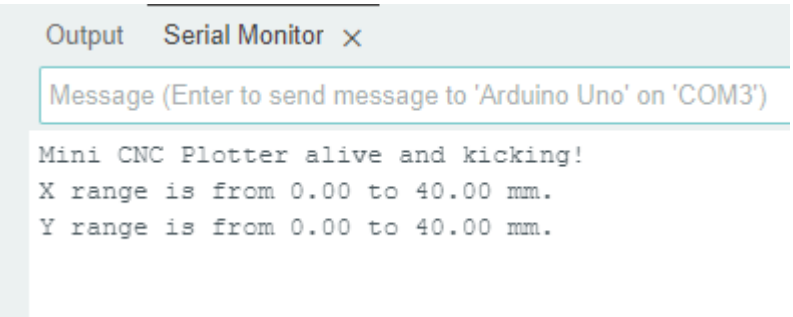
Funkcija *delay(LineDelay)* koristi se kako bi se pauziralo izvođenje programa na određeno vrijeme, što omogućava stabilizaciju alata prije nastavka. Nakon toga, varijable Xpos i Ypos se ažuriraju sa novim vrijednostima x1 i y1, što označava da je alatni stroj prešao na nove koordinate. Ova ažuriranja su važna kako bi se osiguralo da program održava ispravne informacije o trenutnoj poziciji alatnog stroja.

```
void penUp() {
  penServo.write(penZUp);
  delay(penDelay);
  Zpos=Zmax;
  digitalWrite(15, LOW);
    digitalWrite(16, HIGH);
  if (verbose) {
    Serial.println("Pen up!");

  }
}
void penDown() {
  penServo.write(penZDown);
  delay(penDelay);
  Zpos=Zmin;
  digitalWrite(15, HIGH);
    digitalWrite(16, LOW);
  if (verbose) {
    Serial.println("Pen down.");


  }
}
```

Funkcija *penUp()* podiže alatni stroj koristeći servomotor, postavlja Zpos na maksimalnu vrijednost, a digitalne izlaze na odgovarajuće stanje. Ako je varijabla *verbose* postavljena na *true*, ispisuje se poruka "Pen up!" na serijskoj konzoli. Slično

tome, funkcija *penDown()* spušta alatni stroj koristeći servomotor, postavlja Zpos na minimalnu vrijednost i ažurira digitalne izlaze, te ako je *verbose* postavljeno na *true*, ispisuje poruku "Pen down." na serijskoj konzoli.

Nakon učitavanja koda u arduino pločicu , serial monitor ispisuje poruku prikazanu na slici 24. Ispisana poruka označava ispravnost učitanog koda.



Slika 24: Ispis poruke nakon učitavanja koda

Izvor: Autor

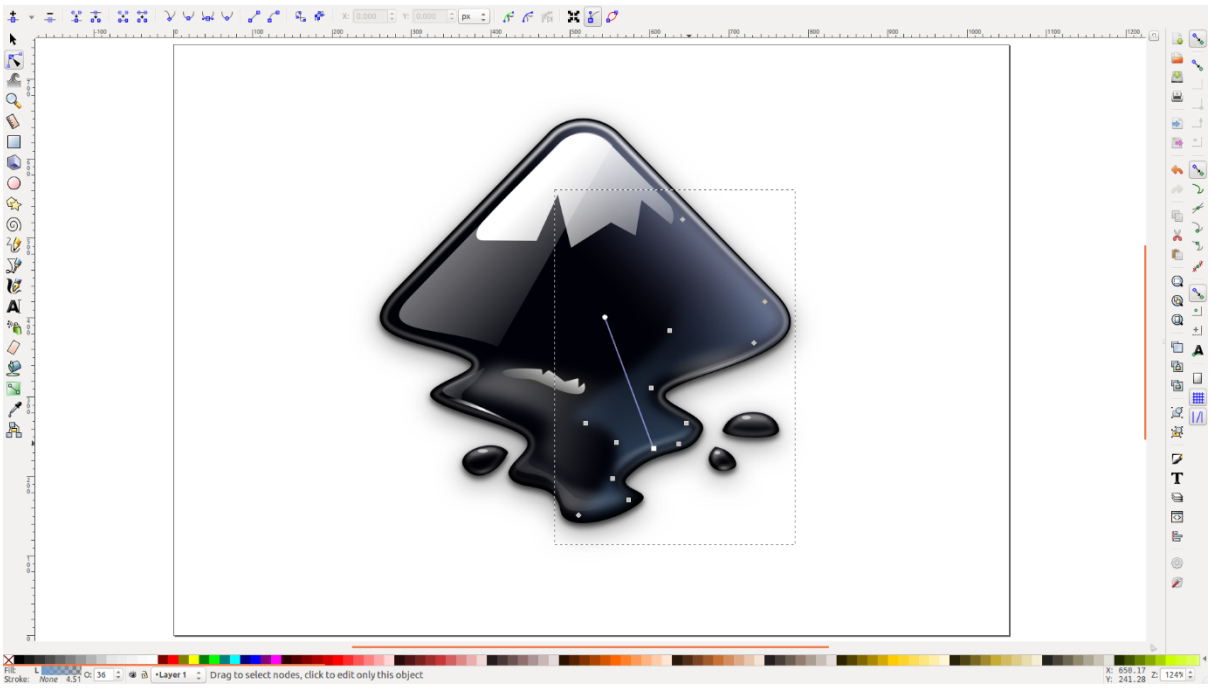
4.2 SOFTVER ZA IZRADU SLIKA ZA PRINTANJE

G-code[1] je jezik CNC stroja. U ovom projektu koristi se Inkscape softver i MakerBot G-code knjižnicu kako bi se generirao G-code slike. Inkscape je besplatan i open-source vektorski grafički softver koji omogućava dizajniranje i uređivanje vektorskih grafika.

Inkscape je softver koji se fokusira na vektorsku grafiku, što znači da slike gradi od vektora (linija, krugova, poligona) umjesto od piksela. To omogućava skaliranje slike bez gubitka kvalitete. Potpuno je besplatan i open-source, što znači da je izvorni kod otvoren za javnost i dostupan svima bez potrebe za plaćanjem licence. Softver nudi raznovrsne alate za crtanje, uključujući olovku, četku, geometrijske oblike te alate za manipulaciju objektima poput selektora i transformacija. Inkscape podržava uvoz i izvoz različitih formata datoteka, uključujući SVG (Scalable Vector Graphics), AI

[1] https://support.makerbot.com/s/article/1667337572861

(Adobe Illustrator), PDF i druge popularne formate. Omogućava dodavanje i uređivanje teksta u različitim fontovima, stilizaciju linija i oblika, primjenu gradijenta, filtera i efekata. Koristi sistem slojeva koji omogućava organizaciju i uređivanje slika na više nivoa, što je korisno za kompleksne projekte gdje je potrebno održavati preglednost. Inkscape ima aktivan korisnički forum i široku dokumentaciju koja olakšava učenje i rješavanje problema. Dostupan je za različite operativne sustave, uključujući Windows, macOS i Linux. (Inkscape Project, Software Freedom Conservancy, Inc., bez datuma)

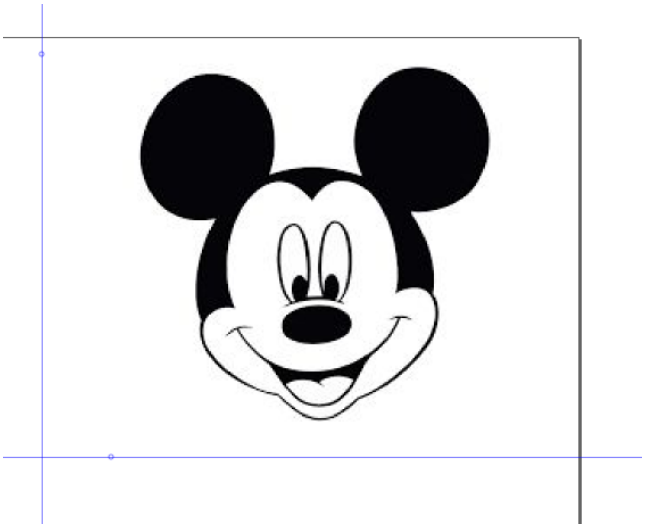


Slika 25: Inscape softversko sučelje

Izvor: https://www.maker-works.com/class-software-inkscape-102

Preuzeto: 28.03.2024.

Pomoću Inscape softvera potrebno je izraditi sliku za ispis na CNC printeru. Odabrana slika skalirana je na dimenzije ne veće od 40 mm x 40 mm. Te dimenzije odgovaraju maksimalnom hodu koračnog motora CNC printera.



Slika 26: Odabrana slika za ispis

Izvor: Autor

4.3 SOFTVER ZA UPRAVLJANJE CNC PRINTERA

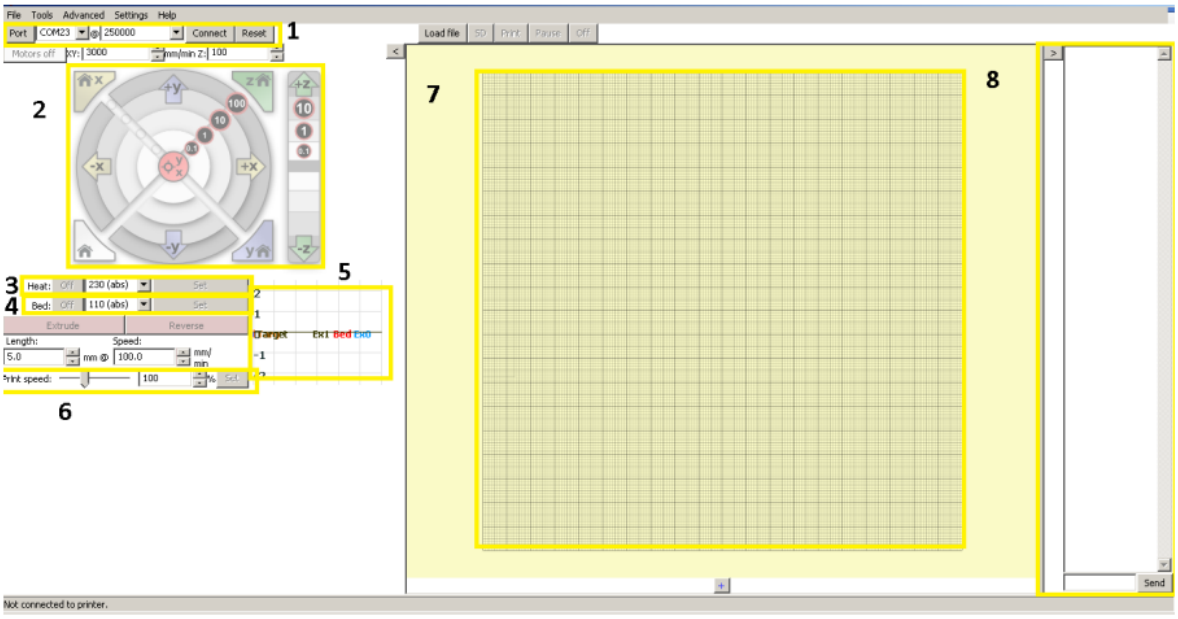
Pronterface je besplatan i open-source softver dizajniran za upravljanje i kontrolu 3D printera povezanih s računarom. Pronterface je softver koji omogućava korisnicima praćenje i upravljanje parametrima 3D printera u stvarnom vremenu. Možete pratiti temperaturu ekstrudera i radne ploče, kao i zaustavljanje ili pauziranje ispisivanja po potrebi. Softver ima jednostavno grafičko korisničko sučelje koje omogućava lako praćenje statusa ispisa, postavljanje parametara i kontrolu uređaja. Kompatibilan je s različitim 3D printerima i može se koristiti s različitim firmware-ima, uključujući Marlin, Repetier i slične. Korisnici mogu prilagoditi različite postavke ispisa, kao što su brzina, temperatura, rezolucija i druge karakteristike prema svojim potrebama. Pronterface omogućava ručnu kontrolu osi, što korisnicima omogućava postavljanje i testiranje pozicija steper motora prije početka ispisa. Također omogućava praćenje napretka ispisa, a korisnici mogu reagirati na eventualne probleme, zaustaviti ispisivanje ili prilagoditi postavke kako bi dobili željeni rezultat. Pronterface je potpuno besplatan, a izvorni kod je otvoren za javnost, što znači da je dostupan svima bez potrebe za plaćanjem licence (ALL3DP, pronterface, bez datuma).

„Program Pronterface moguće je koristiti na Windows i Linux platformi, a u radu je korišten na Windows 10 platformi. Napisan je u programskom jeziku Python. U

programu se klikne na gumb Load File te se s računala odabere model koji je prethodno izrađen u programu za crtanje te program automatski taj model pretvara u G-code. To je kod koji je razumljiv kontrolnoj elektronici samog pisača tj. za numeričko upravljanje. G-code je programski jezik u kojemu korisnik „govori“ stroju kako da nešto izradi.

Primjer G-code: G1 X90.6 Y13.8 E22.4 (kreći se 90.mm po X osi i 13.8 mm po Y osi te ispiši 22.4 mm plastike). Kada program završi s kompajliranjem pisač se automatski kalibrira (stavlja u početni položaj).“ (Uršulin, 2015, str. 19).

Najznačajnije funkcije Pronterface softvera označene su na slici 27 i objašnjene su u daljnjem djelu teksta.



Slika 27: Pronterface softversko sučelje

Izvor: https://zir.nsk.hr/islandora/object/unin:141/preview

Preuzeto: 28.03.2024.

„1. U ovom prozoru odabire se PORT na kojem se spaja CNC pisač, broj bitova koji de se slati te dva gumba: CONNECT koji služi da se selektirani port otvori te započne konekcija CNC pisača s računalom te RESET koja vraća sve postavke na zadane.

2. Ovaj prozor služi da se koračni motori kalibriraju te postave na početne pozicije koje definiraju ranije spominjani krajnici.

3. Postavljanje temperature grijača: može se unaprijed odabrati zadana temperatura za PLA i ABS plastiku u padajućem izborniku. (Za 3D printer i rad s plastikom)

4. Postavljanje temperature grijače ploče. (Za 3D printer i rad s plastikom)

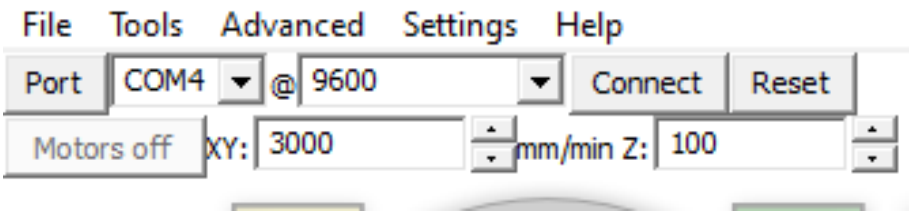
5. Graf koji pokazuje temperaturu u ovisnosti o vremenu.

6. Podešavanje brzine ispisa.

7. Prostor u kojem se iscrtava željeni model te kako de oni izgledati.

8. U ovom prozoru moguće je poslati komande pisaču koje su napisane u G-code“ (Uršulin, 2015, str. 20).

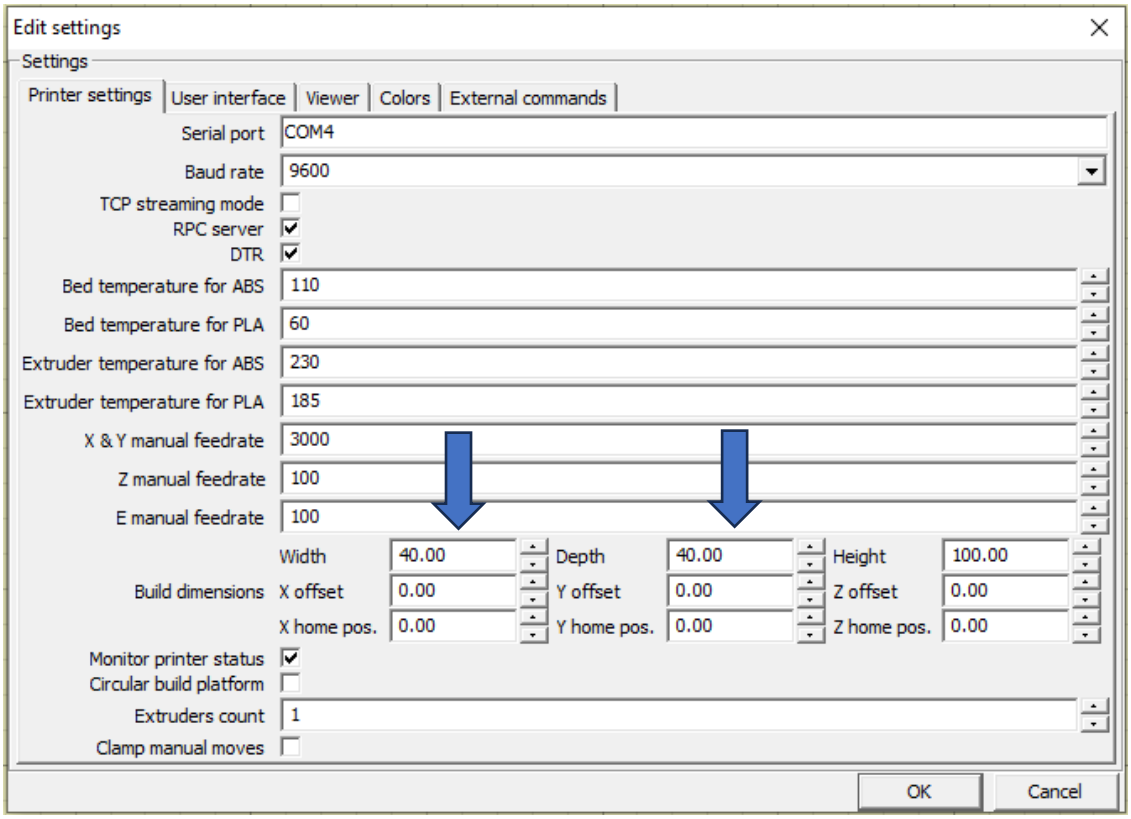
Za potrebe ovog rada postavljeni su „Port“ na COM4 (USB priključak na koji je spojen Arduino), te brzina prijenosa podataka na 9600. Postavljene vrijednosti su prikazane na slici 28. Nakon unosa zadanih vrijednosti odabire se tipka „Connect“. Na taj se način povezuje Arduino pločica s Pronterface softverom.



Slika 28: Povezivanje arduino pločice i Pronterface softvera

Izvor: Autor

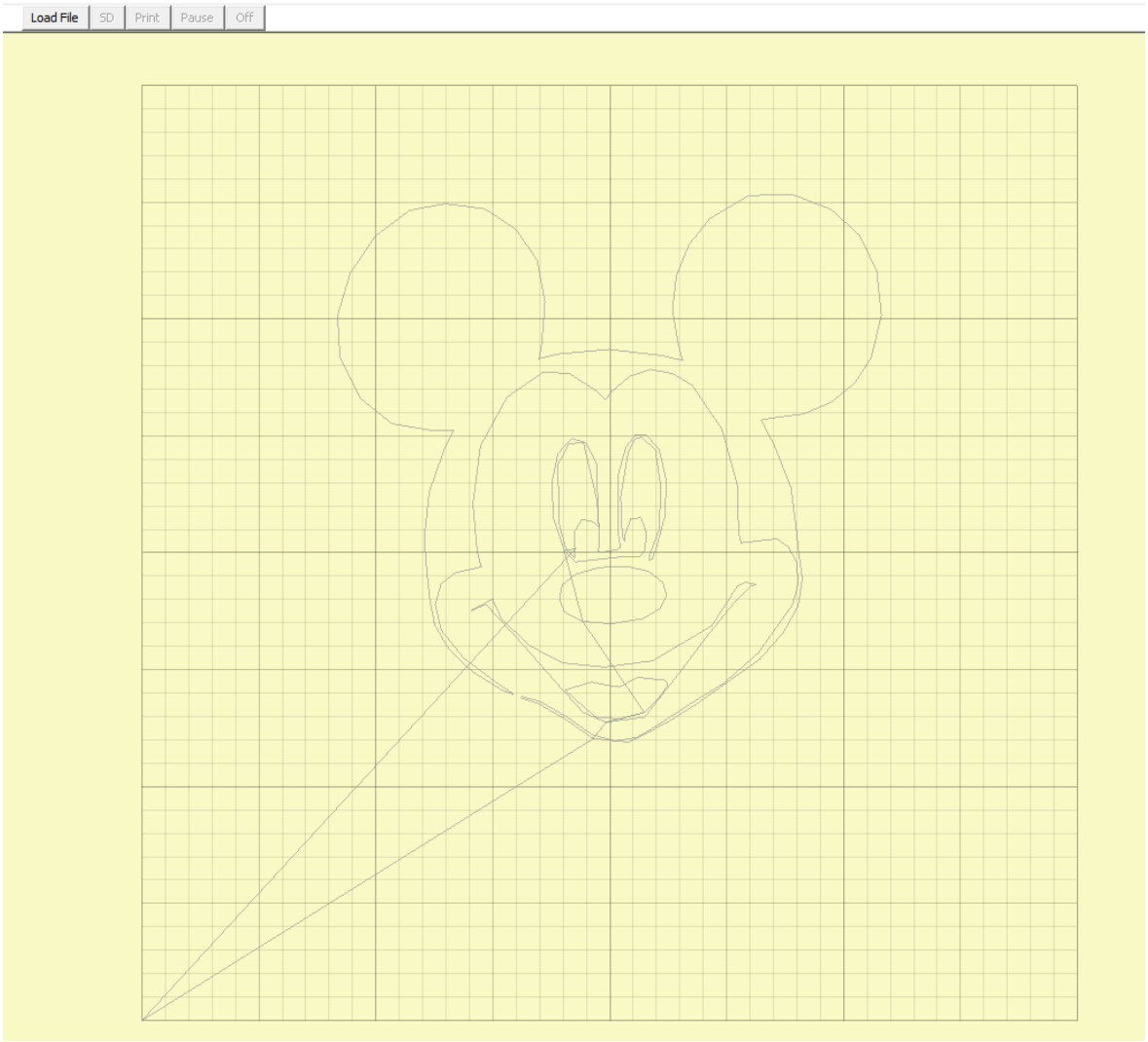
U sljedećem koraku potrebno je postaviti granice ispisa u postavkama softvera Pronterface. Zbog ograničenog hoda koračnog motora potrebno je upisati 40 mm u zadane prozore označene na slici 29.



Slika 29: Postavke ispisa u Pronterface softveru.

Izvor: Autor

Posljednji korak prije ispisa zahtijeva učitavanje odabrane sličice u Pronterface softver. Pritiskom na gumb „load file“ otvara se prozor te odabire se g-code datoteka odabrana za ispis.



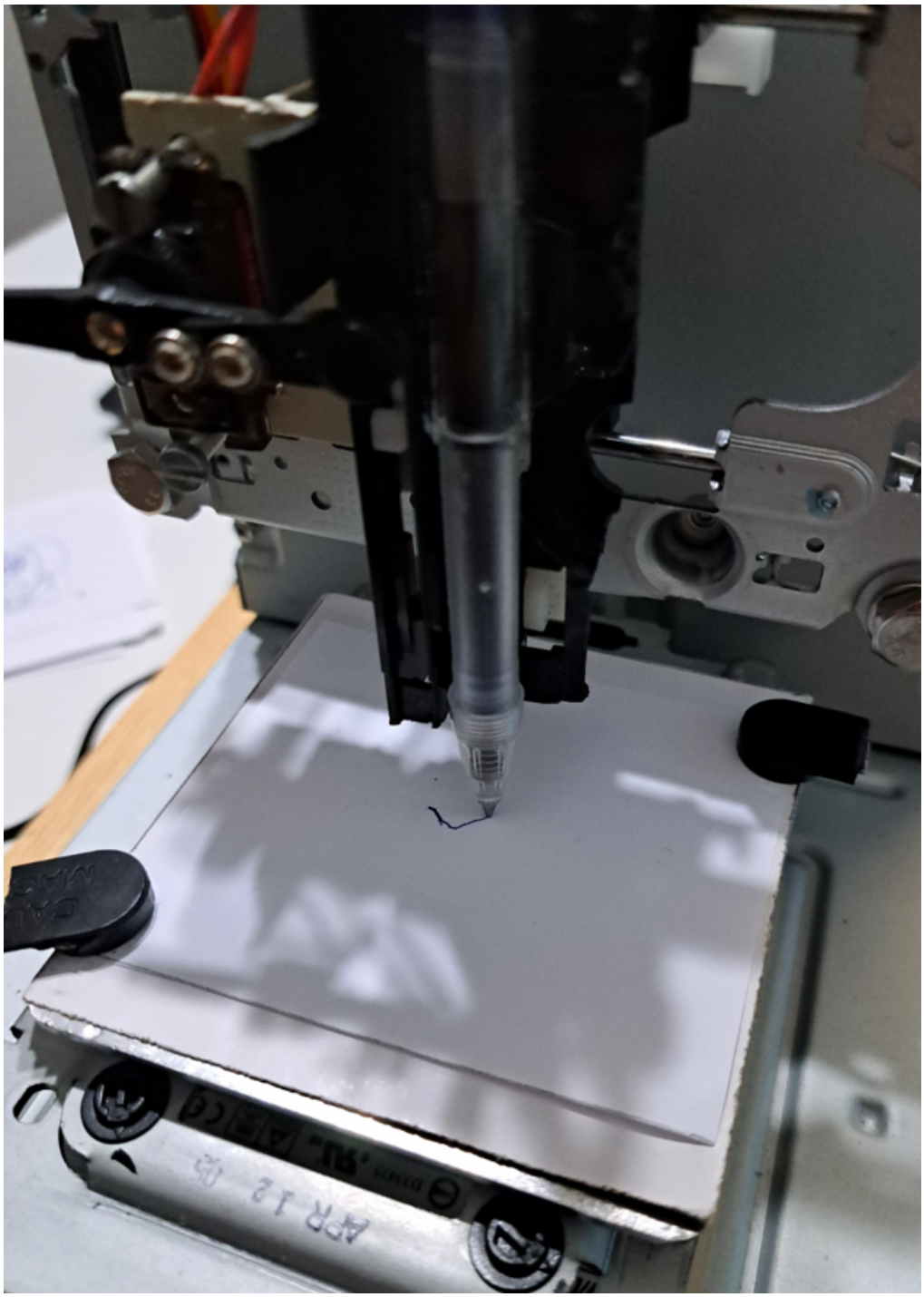
Slika 30: Učitavanje slike u Pronterface softveru

Izvor: Autor

Po završetku učitavanja slike u softver potrebno je stisnuti tipku „print“ i na taj način započeti će rad printera i ispis slike na papirnatu podlogu.

5. DISKUSIJA

Praćenjem rada printera utvrđene su određene neispravnosti (Slika 31 i Slika 32) koje su dodatno istražene.

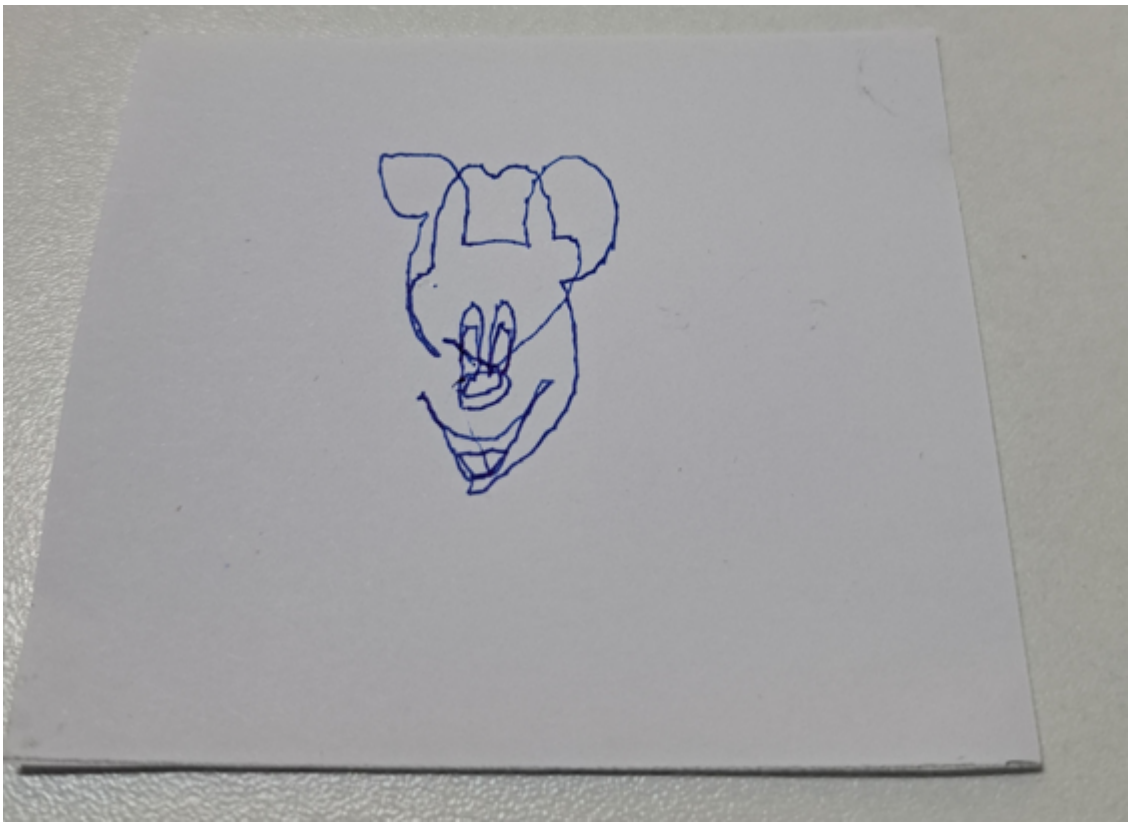


Slika 31: Početak ispisa slike

Izvor. Autor

Slika 32: Praćenje rada printera

Izvor: Autor



Slika 33: Rezultat ispisa printera

Izvor: Autor

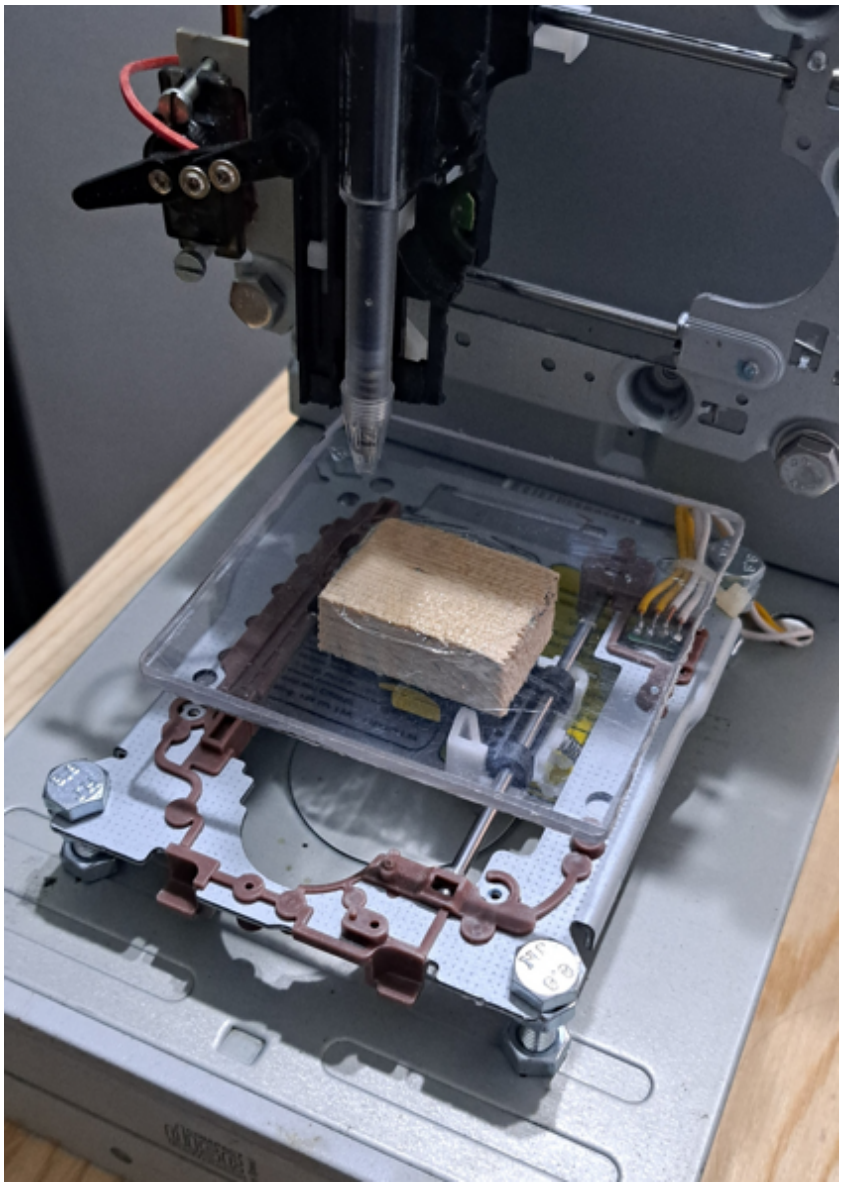
Iz rezultata na slici 33, vidljivo je odstupanje u ispisu slike od učitane slike u softver Pronterface.

Dodatnom provjerom ispitalo se rade li koračni motori ispravno i izvršila se provjera postavljenih granica za ispis, te korigiralo vrijednosti u slučaju odstupanja podataka.

Rezultati ispisa ostaju isti prilikom promjene jačine napona sa 7.5 V na 9 V što je ukazalo na mogući neispravan rad koračnog motora.

Uvidom u rezultate ispisa te dodatnom kontrolom motora, utvrđena je neispravnost u kretnjama vodoravnog koračnog motora. Motor koji ima zadatak pomicati pločicu naprijed - natrag pri kretanju preskače korake i vibrira. Zbog toga slika 33. iz rezultata prikazuje nedostatke u radu CNC printera.

Sljedeći korak u ispravljanju nedostatka je zamjena koračnog motora.



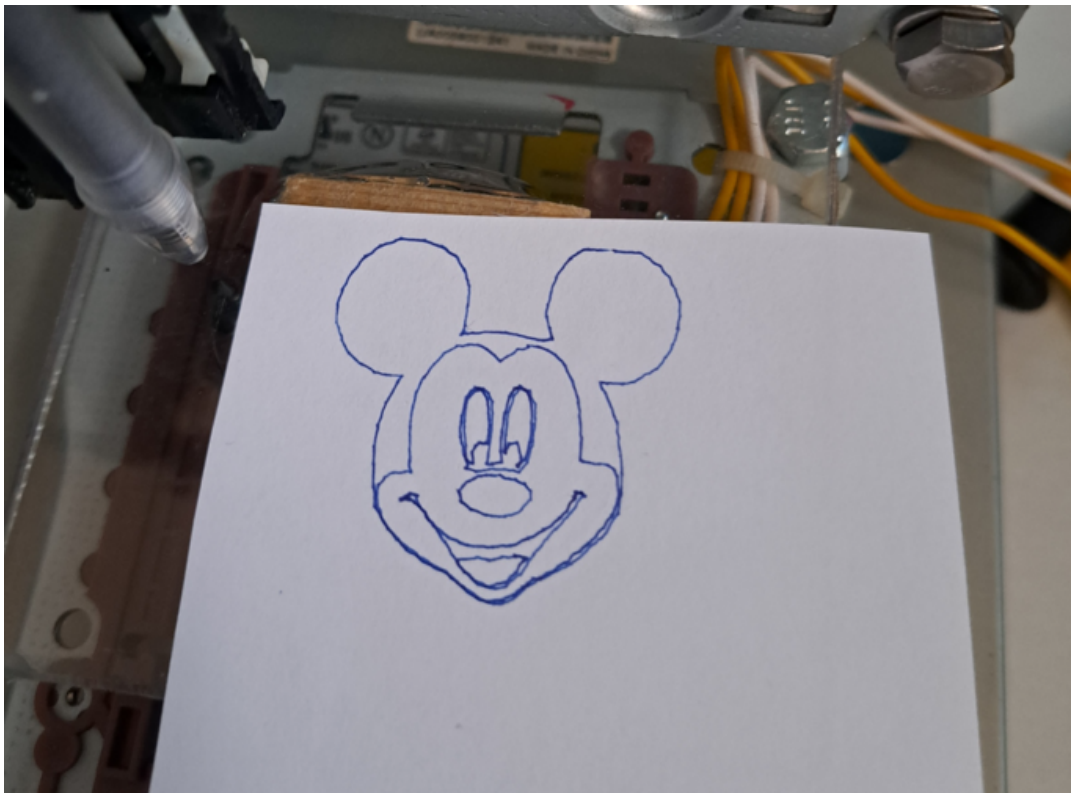
Slika 34: Novi koračni motor i plastična pločica za podlogu

Izvor: Autor

Zbog lakšeg rada printera, osim samog koračnog motora zamijenjena je i pločica za pisanje. Aluminijska pločica zamijenjena je plastičnom. Zbog lakšeg materijala

smanjuje se opterećenje na koračni motor i olakšava njegov rad i kretanje naprijed – natrag.

Nakon zamjene koračnog motora i spajanja električnih žica na L293D modul, pokreće se CNC printer i testira se ispisivanje zadane slike.



Slika 35: Rezultat ispisa nakon zamjene neispravnog koračnog motora

Izvor: Autor

Ispravno ispisana slika ukazuje na pravilan rad svih dijelova CNC printera i uspješnost u izradi projekta.

6. ZAKLJUČAK

Mini CNC 2D printer izrađen iz reciklirane elektronike omogućava korisnicima eksperimentiranje i istraživanje numeričke kontrolne obrade programiranjem, pružajući jednostavan, ali efikasan način za crtanje na ravnoj površini.

Komponente poput stepper motora, Arduino Uno, L293D Motor Shield modula i servomotora Tower Pro 9G čine temelj ovog projekta, dok se kemijska olovka koristi za crtanje po papiru. Reciklirana DVD kućišta služe kao osnova za postolje CNC uređaja, dodajući održivost i ekološki aspekt projektu.

Električna shema i Arduino kod pružaju kontrolu nad motorima, omogućujući precizno pozicioniranje i crtanje. Kroz proces izrade i spajanja komponenata, korisnici stvaraju funkcionalan CNC mini printer sposoban za ispis različitih sličica.

Ovaj projekt također naglašava važnost recikliranja elektronike i kreativnog pristupa tehnologiji. Kroz kombinaciju tehničkih vještina, programiranja i upravljanja motorima, stvara se prilika za učenje i zabavu, istražujući granice mogućnosti „uradi sam“ pristupa CNC tehnologiji.

LITERATURA

1. 3ERP (bez datuma). G-code For CNC Machine. Preuzeto 15.lipnja.2024 s https://www.3erp.com/blog/g-code-for-cnc/
2. ALL3DP (bez datuma). Pronterface. Preuzeto 14.lipnja.2024 s https://all3dp.com/2/pronterface-how-to-download-install-and-set-it-up/
3. Amazon (bez datuma). TowerPro SG 90 micro servo motor. Preuzeto 15. ožujka 2024. s https://www.amazon.in/Robodo-Electronics-Tower-Micro-Servo/dp/B00MTFFAE0?th=1
4. Amazon (bez datuma). Kabel za Arduino. Preuzeto 18.ožujka 2024. s https://www.amazon.in/Robu-Feet-Cable-Arduino-MEGA/dp/B074FX5M67
5. Arduobotics (bez datuma). DC konektor. Preuzeto 15. ožujka 2024. s https://ardubotics.eu/hr/ostalo/1807-dc-zenski-konektor-za-napjanje-pcb-5521mm-dc-005.html
6. Blažević, Z. (2004). Programiranje CNC tokarilice i glodalice. Učenje CNC programiranja na tokarilici i glodalici za srednje škole.
7. Engineers garage (bez datuma). Arduino L293D motor shield modul. Preuzeto 15. ožujka 2024. s https://www.engineersgarage.com/arduino-l293d-motor-driver-shield-tutorial/
8. Exploring Arduino (bez datuma). Jumper wires. Preuzeto 15. ožujka 2024. s https://www.exploringarduino.com/parts/jumper-wires/
9. Inkscape Project (bez datuma). Software Freedom Conservancy, Inc. Preuzeto 16.lipnja.2024. s https://inkscape.org/about/
10. Kult web-shop (bez datuma). Kemijska olovka. Preuzeto 18.ožujka 2024. s https://www.kult-vk.hr/pisaci-pribor-kemijske-ol-roleri-markeri-tehnolovke-grafitne-olovke/13-kemijska-olovka-fornax-f-070-gumirana-3856000458757.html
11. Labdomotic (bez datuma). Električna shema. Preuzeto 18.ožujka 2024. s https://www.labdomotic.com/2019/04/30/youtube-guida-software-cnc-diy/
12. Maker Works (bez datuma). Inkscape softver. Preuzeto 28.ožujka 2024. s https://www.maker-works.com/class-software-inkscape-102

13. Pabolu, V., Srinivas, S. (2010). Design and Implementation of a Three Dimensional CNC Machine. International Journal on Computer Science and Engineering.
14. Pevex (bez datuma). Matični vijak s maticom. Preuzeto 18.ožujka 2024. s https://pevex.hr/maticni-vijak-s-maticom-m6x50mm-din931-25-1
15. Pezer D. (2022). Programiranje CNC strojeva Izvor: https://www.oss.unist.hr/Portals/0/adam/Contents/TnPnQrlyXEeYJna9tPx2wQ/Text/PROGRAMIRANJE%20CNC%20STROJEVA%20Sinumerik%20840D.pdf
16. SparkFun Electronics from Boulder, USA(bez datuma). Arduino Uno - R3. Preuzeto 01.07.2024. s https://www.flickr.com/photos/sparkfun/8406865680/
17. Svijet medija (bez datuma). Strujni adapter GOOBAY. Preuzeto 18.ožujka 2024. s https://www.svijet-medija.hr/art/strujni-adapter-goobay-59030-3-12v-dc-2250ma-100-240v-univerzalni/88767
18. Pandian, S., Pandian, S. (2014). A LOW-COST BUILD-YOUR-OWN THREE AXIS CNC MILL PROTOTYPE. International Journal on Mechanical Engineering and Robotics (IJMER).
19. Uršulin Ž.(2015). Izrada i testiranje modela 3D pisača baziranog na Arduino platformi. Preuzeto 26. travnja 2024. s https://zir.nsk.hr/islandora/object/unin:141/preview

POPIS SLIKA

PRILOG – programski kod za Arduino

```
#include <Servo.h>
#include <AFMotor.h>

#define LINE_BUFFER_LENGTH 512

char STEP = MICROSTEP ;

const int penZUp = 95;
const int penZDown = 83;

const int penServoPin =10 ;

const int stepsPerRevolution = 48;

Servo penServo;

AF_Stepper myStepperY(stepsPerRevolution,1);
AF_Stepper myStepperX(stepsPerRevolution,2);

struct point {
  float x;
  float y;
  float z;
};

struct point actuatorPos;

float StepInc = 1;
int StepDelay = 0;
int LineDelay =0;
int penDelay = 50;

float StepsPerMillimeterX = 100.0;
float StepsPerMillimeterY = 100.0;

float Xmin = 0;
float Xmax = 40;
float Ymin = 0;
float Ymax = 40;
float Zmin = 0;
float Zmax = 1;

float Xpos = Xmin;
float Ypos = Ymin;
```

```
float Zpos = Zmax;


boolean verbose = false;

void setup() {

  Serial.begin( 9600 );

  penServo.attach(penServoPin);
  penServo.write(penZUp);
  delay(100);


  myStepperX.setSpeed(600);

  myStepperY.setSpeed(600);


  Serial.println("Mini CNC Plotter alive and kicking!");
  Serial.print("X range is from ");
  Serial.print(Xmin);
  Serial.print(" to ");
  Serial.print(Xmax);
  Serial.println(" mm.");
  Serial.print("Y range is from ");
  Serial.print(Ymin);
  Serial.print(" to ");
  Serial.print(Ymax);
  Serial.println(" mm.");
}


void loop()
{

  delay(100);
  char line[ LINE_BUFFER_LENGTH ];
  char c;
  int lineIndex;
  bool lineIsComment, lineSemiColon;

  lineIndex = 0;
  lineSemiColon = false;
  lineIsComment = false;

  while (1) {
```

```
while ( Serial.available()>0 ) {
  c = Serial.read();
  if (( c == '\n') || (c == '\r') ) {
    if ( lineIndex > 0 ) {
      line[ lineIndex ] = '\0';
      if (verbose) {
        Serial.print( "Received : ");
        Serial.println( line );
      }
      processIncomingLine( line, lineIndex );
      lineIndex = 0;
    }
    else {

    }
    lineIsComment = false;
    lineSemiColon = false;
    Serial.println("ok");
  }
  else {
    if ( (lineIsComment) || (lineSemiColon) ) {
      if ( c == ')' )  lineIsComment = false;
    }
    else {
      if ( c <= ' ' ) {
      }
      else if ( c == '/' ) {
      }
      else if ( c == '(' ) {
        lineIsComment = true;
      }
      else if ( c == ';' ) {
        lineSemiColon = true;
      }
      else if ( lineIndex >= LINE_BUFFER_LENGTH-1 ) {
        Serial.println( "ERROR - lineBuffer overflow" );
        lineIsComment = false;
        lineSemiColon = false;
      }
      else if ( c >= 'a' && c <= 'z' ) {
        line[ lineIndex++ ] = c-'a'+'A';
      }
      else {
        line[ lineIndex++ ] = c;
      }
    }
  }
```

```
    }
  }
}

void processIncomingLine( char* line, int charNB ) {
  int currentIndex = 0;
  char buffer[ 64 ];
  struct point newPos;

  newPos.x = 0.0;
  newPos.y = 0.0;


  while( currentIndex < charNB ) {
    switch ( line[ currentIndex++ ] ) {
    case 'U':
      penUp();
      break;
    case 'D':
      penDown();
      break;
    case 'G':
      buffer[0] = line[ currentIndex++ ];

      buffer[1] = '\0';

      switch ( atoi( buffer ) ){
      case 0:
      case 1:

        char* indexX = strchr( line+currentIndex, 'X' );
        char* indexY = strchr( line+currentIndex, 'Y' );
        if ( indexY <= 0 ) {
          newPos.x = atof( indexX + 1);
          newPos.y = actuatorPos.y;
        }
        else if ( indexX <= 0 ) {
          newPos.y = atof( indexY + 1);
          newPos.x = actuatorPos.x;
        }
        else {
          newPos.y = atof( indexY + 1);
          indexY = '\0';
          newPos.x = atof( indexX + 1);
        }
        drawLine(newPos.x, newPos.y );

        actuatorPos.x = newPos.x;
```

```
        actuatorPos.y = newPos.y;
        break;
      }
      break;
    case 'M':
      buffer[0] = line[ currentIndex++ ];
      buffer[1] = line[ currentIndex++ ];
      buffer[2] = line[ currentIndex++ ];
      buffer[3] = '\0';
      switch ( atoi( buffer ) ){
      case 300:
        {
          char* indexS = strchr( line+currentIndex, 'S' );
          float Spos = atof( indexS + 1);

          if (Spos == 30) {
            penDown();
          }
          if (Spos == 50) {
            penUp();
          }
          break;
        }
      case 114:
        Serial.print( "Absolute position : X = " );
        Serial.print( actuatorPos.x );
        Serial.print( "  -  Y = " );
        Serial.println( actuatorPos.y );
        break;
      default:
        Serial.print( "Command not recognized : M");
        Serial.println( buffer );
      }
    }
  }


}


void drawLine(float x1, float y1) {

  if (verbose)
  {
    Serial.print("fx1, fy1: ");
    Serial.print(x1);
    Serial.print(",");
```

```
  Serial.print(y1);
  Serial.println("");
}


if (x1 >= Xmax) {
  x1 = Xmax;
}
if (x1 <= Xmin) {
  x1 = Xmin;
}
if (y1 >= Ymax) {
  y1 = Ymax;
}
if (y1 <= Ymin) {
  y1 = Ymin;
}

if (verbose)
{
  Serial.print("Xpos, Ypos: ");
  Serial.print(Xpos);
  Serial.print(",");
  Serial.print(Ypos);
  Serial.println("");
}

if (verbose)
{
  Serial.print("x1, y1: ");
  Serial.print(x1);
  Serial.print(",");
  Serial.print(y1);
  Serial.println("");
}


x1 = (int)(x1*StepsPerMillimeterX);
y1 = (int)(y1*StepsPerMillimeterY);
float x0 = Xpos;
float y0 = Ypos;


long dx = abs(x1-x0);
long dy = abs(y1-y0);
int sx = x0<x1 ? StepInc : -StepInc;
int sy = y0<y1 ? StepInc : -StepInc;
```

```
long i;
long over = 0;

if (dx > dy) {
  for (i=0; i<dx; ++i) {
    myStepperX.onestep(sx,STEP);
    over+=dy;
    if (over>=dx) {
      over-=dx;
      myStepperY.onestep(sy,STEP);
    }
  delay(StepDelay);
  }
}
else {
  for (i=0; i<dy; ++i) {
    myStepperY.onestep(sy,STEP);
    over+=dx;
    if (over>=dy) {
      over-=dy;
      myStepperX.onestep(sx,STEP);
    }
    delay(StepDelay);
  }
}

if (verbose)
{
  Serial.print("dx, dy:");
  Serial.print(dx);
  Serial.print(",");
  Serial.print(dy);
  Serial.println("");
}

if (verbose)
{
  Serial.print("Going to (");
  Serial.print(x0);
  Serial.print(",");
  Serial.print(y0);
  Serial.println(")");
}


delay(LineDelay);

Xpos = x1;
```

```
  Ypos = y1;
}

void penUp() {
  penServo.write(penZUp);
  delay(penDelay);
  Zpos=Zmax;
  digitalWrite(15, LOW);
    digitalWrite(16, HIGH);
  if (verbose) {
    Serial.println("Pen up!");

  }
}
void penDown() {
  penServo.write(penZDown);
  delay(penDelay);
  Zpos=Zmin;
  digitalWrite(15, HIGH);
    digitalWrite(16, LOW);
  if (verbose) {
    Serial.println("Pen down.");


  }
}
```