SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Seminarski zadatak

Praktikum robotike

LEGO ROBOART

Tehnička dokumentacija

Grupa G:

Anela Omerović
Gordan Ovčarić
Una Pale
Goran Popović

Zagreb, lipanj 2014.

Sadržaj

1. Uvod

1.1. Zadatak

Cilj seminara je izrada specijaliziranog slikarskog robota, odnosno robota koji će biti sposoban precrtati zadanu sliku. Sustav se sastoji od troosnog TRR robota s kistom u ulozi alata, originalne slike koju treba preslikati, kamere pozicionirane iznad nje, bijelog platna, te kantice s bojom koji se nalazi unutar radnog prostora robota.

Projekt je podijeljen u dvije faze; fazu virtualnog projektiranja i mehaničku izradu i projektiranje stvarnog robota. U prvoj fazi bilo je potrebno izraditi virtualni model robota prema specifikacijama:

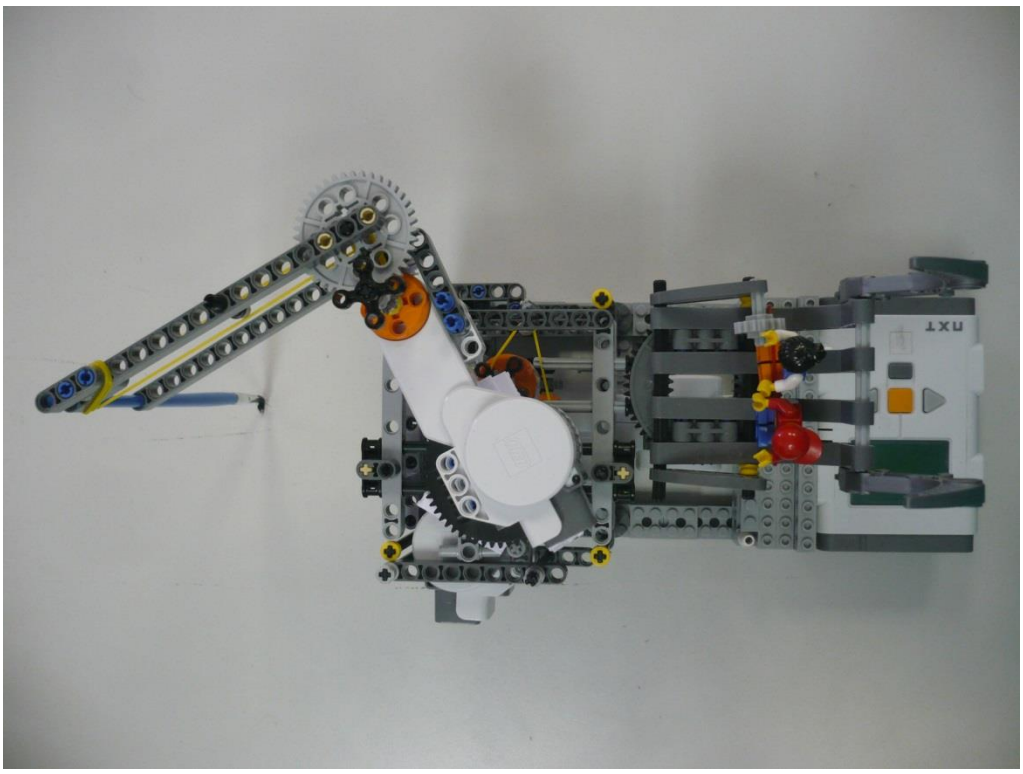
- maksimalni dohvat u horizontalnoj ravnini: 0.2 m
- minimalni hod u vertikalnoj ravnini: ±0.05 m

Također bilo je potrebno napraviti grafičko sučelje za upravljanje u Matlabu, u kojem je bilo nužno omogućiti upravljanje u prostoru varijabli zglobova, te upravljanje robotom u koordinatnom sustavu alata.

U drugoj fazi trebalo je izraditi robot te prilagoditi sustav upravljanja za njega. Također bilo je potrebno implementirati analizu i obradu slike koja će omogućiti robotu prepoznavanje vrsta i pozicija zadanih oblika na originalnoj slici kako bi ih precrtao.

1.2. Općenito

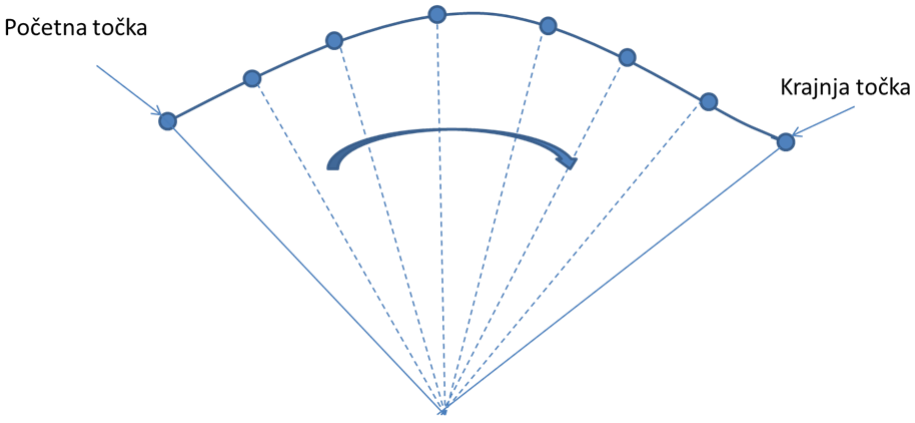
„Roboart“ robot se po kvalifikaciji svrstava u zabavne robote. To je robotska ruka TRR konfiguracije. Sastavljen je iz LEGO Mindstorm NXT kompleta. Zglobovi su mu rotacijski i ostvareni su pomoću servomotora i zupčanika. Upravljanje servomotorima se ostvaruje preko računala uz pomoć NXT kutije koja se priključuje na USB ulaz računala. Naredbe koje se zadaju robotskoj ruci zadaju se u napravljenom grafičkom sučelju. Robotu je omogućeno micanje pojedinih zglobova, povlačenje linije i crtanje kružnice.
Uz upravljanje robotom, obradom slike snimljene kamerom, omogućeno je prepoznavanje kružnica i linija koje su nacrtane na papiru. Nakon toga robot može prepoznate linije i kružnice, uz pomoć kista i boje nacrtati na papiru.



Slika 1. Robotska ruka „Roboart“ slikana odozgo

1.3. Ideja upravljanja virtualnim modelom

Virtualni model robota napravljen je u Matlabu, kako bi se testiralo upravljanje prije same konstrukcije robota. Kada blok „VR Model“ u Simulinku dobije signale na ulaz on trenutno proslijedi te informacije na dijelove modela. Kako pomicanje robotske ruke u simulaciji ne bi bilo naoko diskretno, trajektorija robota se podijelila u više manjih dijelova kroz koje bi dijelovi ruke prolazili i tako davali prividnu kontinuiranost pokreta.

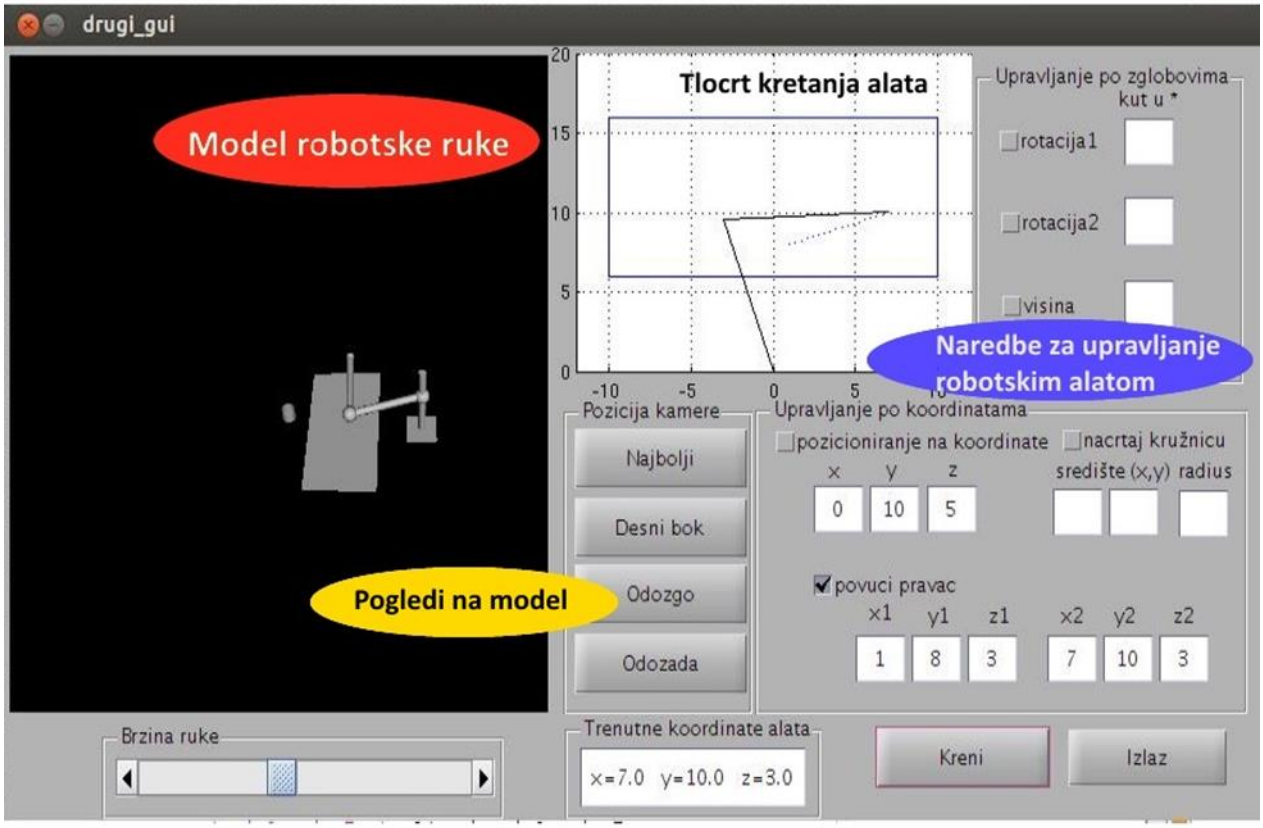


Slika 2. Opis kretanja od početne do konačne točke

2. Virtualni model robotske ruke

2.1. Grafičko sučelje

Pokretanjem skripte drugi_gui.m otvara se prozor u kojem je prikazan virtualni 3D model robotske ruke. Sučelje je podijeljeno na dio za mijenjanje pozicije kamere, dio sa tlocrtom radnog prostora u kojem se ostavlja trag na mjestima kroz koje je prošao vrh alata kako bi se vizualizirana zadana krivulja, te sadrži dio za upravljanje. Kada korisnik želi zadati naredbu robotu mora označiti jednu od opcija za gibanje robota koje su ponuđene, te u bijele kvadratiće uz odabranu opciju unijeti koordinate. Također, kliznom trakom moguće je mijenjati brzinu crtanja zadane krivulje. Nakon odabira potrebno je pritisnuti gumb "Kreni" i tada se simulacija počinje izvršavati. Pri pritisku gumba "Kreni" mora biti odabrana samo jedna opcija jer inače simulacija neće raditi dobro. U sredini donjeg dijela ekrana prikazuju se trenutne koordinate vrha alata robota. Nakon završetka simulacije, iz simulacije se može izaći pritiskom gumba „Izlaz“ i time se zatvara grafičko sučelje, Simulink i virtualni model.

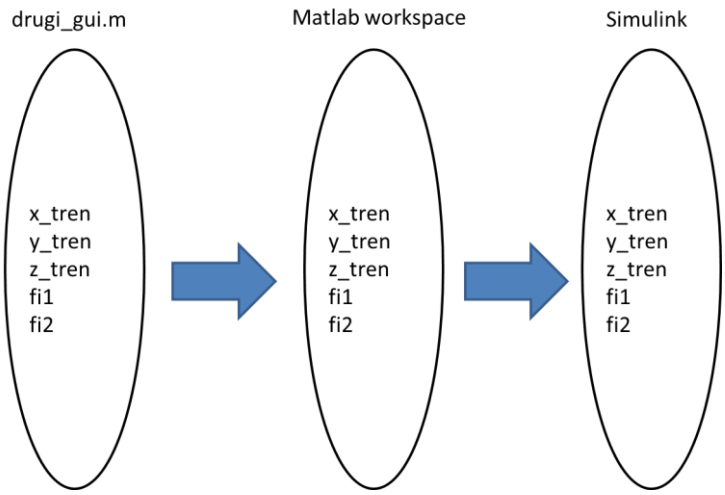


Slika 3. Izgled grafičkog sučelja za virtualnu simulaciju

2.2. Upravljanje virtualnim modelom

Pokretanjem skripte drugi_gui.m osim otvaranja samog grafičkog sučelja, otvara se i virtualni model "jednostavni_robot.wrl", pokreće se Simulink shema "model_ruke.slx". Pritom se inicijaliziraju početne koordinate zglobova i duljine članaka. Koordinate zglobova su varijable koje se pamte kroz cijeli program te su argumenti i izlazi svake funkcije koja ih mijenja.

Pri pritisku gumba "Kreni" odabrana opcija (označena kvačicom) poziva određenu funkciju i prosljeđuje joj unesene vrijednosti. Odabirom pozicioniranja na koordinate poziva se funkcija „pozicioniraj_se.m“ koja iz grafičkog sučelja uzima koordinate na koje treba pozicionirati vrh alata, a također dobiva i trenutne koordinate zglobova. U samoj funkciji se radi direktna kinematika, a zatim se poziva funkcija „odredi_k.m“ koja iz udaljenosti koju mora prijeći vrha alata robotske ruke i brzine, koja se zadaje u grafičkom sučelju, izračunava broj pozicija (koraka) koje će robotska ruka promijeniti između početnog i konačnog položaja (Slika 2). Manjim brojem položaja gibanje će kraće trajati te će simulacija biti brža, te stoga varijabla "brzina" stvarno ima fizičko značenje brzine. Međutim uz veće brzine biti će izraženije diskretno gibanje robotske ruke, a ostavljeni tragovi iscjepkani te neprecizniji.
Nakon određivanja broja diskretnih položaja u varijable x, y i z se zadaju nizovi koordinata koji definiraju zadanu krivulju. U for petlji se iz nizova očitavaju x, y i z koordinata točke u prostoru. Te se koordinate šalju u Simulink gdje se radi inverzna kinematika za dobivene vrijednosti. Vrijednosti koordinata se iz skripti šalju u Matlab-ov workspace, a Simulink u bloku Gain, prije bloka za inverznu kinematiku, uzima vrijednosti x_tren i y_tren iz workspace-a (Slika 4.). Kada se for petlja izvrši, funkcija je gotova. Vrijednosti koje funkcija vraća su trenutne koordinate zglobova. Ista ideja je i kod ostalih naredbi za gibanje robotske ruke.



Slika 4. Prikaz prenošenja vrijednosti iz m-skripti u Simulink

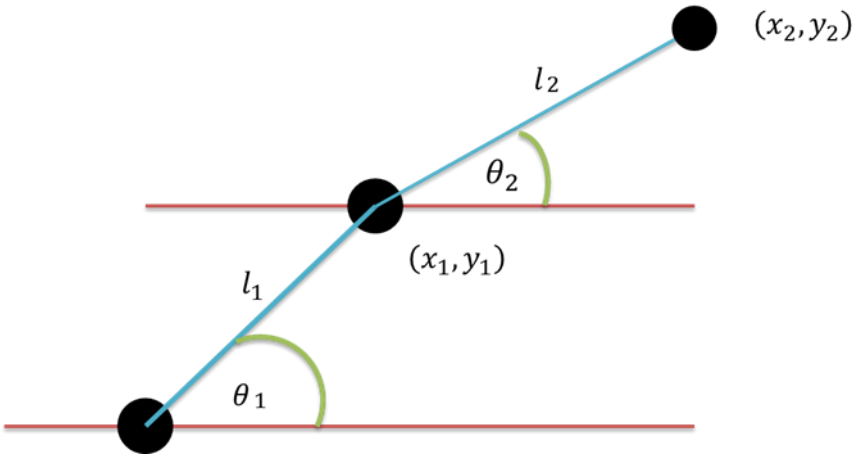
2.2.1 Direktna kinematika

Direktna kinematika služi za pretvaranje kuteva zglobova u pozicije vrha alata. Odnosno za prebacivanje iz sustava zglobova u koordinatni sustav alata. Direktna kinematika je dana forumalama 2.1, 2.2, 2.3, 2.4:

$$x_1 = l_1 * cos\,\theta_1 \qquad (2.1)$$
$$y_1 = l_1 * sin\,\theta_1 \qquad (2.2)$$
$$x_2 = l_1 * cos\,\theta_1 + l_2 * cos\,\theta_2 \qquad (2.3)$$
$$y_2 = l_1 * sin\,\theta_1 + l_2 * sin\,\theta_2 \qquad (2.4)$$

Značenje varijabli prikazano je slikom 5.
Njima se iz trenutnih položaja zglobova i željenih koordinata koje je unio korisnik određuje

udaljenost početne i konačne točke, što se koristi za određivanje broja diskrenih položaja.



Slika 5. Prikaz zglobova i članaka robotske ruke. Te značenje pojedinih varijabli pri direktnoj kinematici

2.2.2. Inverzna kinematika

Inverzna kinematika služi za prebacivanje iz koordinatnog sustava vrha alata u pripadne kutove zglobova. Dana formulama 2.5, 2.6

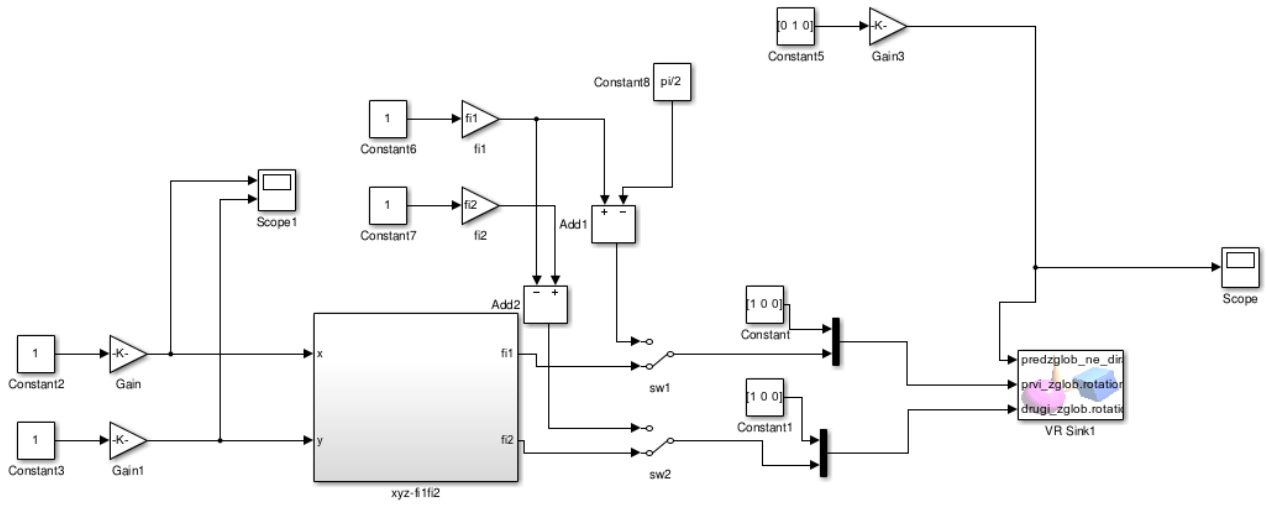
$$\Delta\theta = \theta_1 - \theta_2 = arccos\frac{x^2+y^2-l_1^2-l_2^2}{2*l_1*l_2} \qquad (2.5)$$

$$\theta_2 = atan2(y(l_1\, cos\, \Delta\theta + l_2) - x\, sin\, \Delta\theta\, l1,\ \ l_1\, sin\, \Delta\theta\, y + xl_1\, cos\, \Delta\theta + l_2 x) \qquad (2.6)$$

Značenje varijabli je dano također slikom 5. x i y koordinate vrha alata robotske ruke.
Inverzna kinematika je implementirana u Simulinku u "model_ruke.slx", te u m-funkciji "kutevi.m". Kutovi koje funkcija vraća su apsolutni kutovi, odnosno kutovi u odnosu na koordinatni sustav baze robotske ruke. Kasnije, u upravljanju stvarnim robotom, funkcija "kutevi.m" je prerađena tako da vraća kut drugog članka u odnosu na pomični koordinatni sustav prvog članka.

2.3. Simulink

Iako je veći dio algoritma upravljanja napravljen u m-skriptama, manji dio upravljanja napravljen je u Simulinku. Zbog jednostavnosti slanja signala za kretanje virtualnom modelu taj dio je implementiran u Simulinku. Tu je također napravljen dio za inverznu kinematiku.

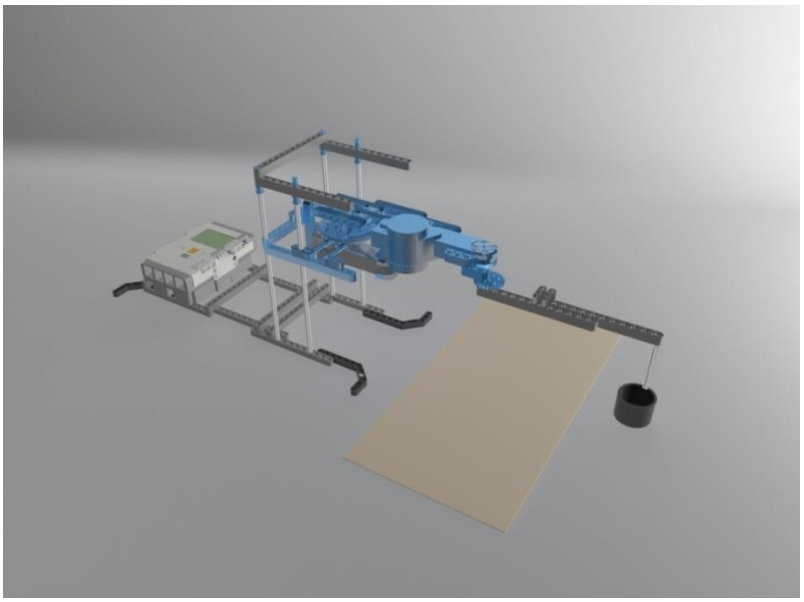


Slika 6. Shema upravljanja u Matlab simulinku

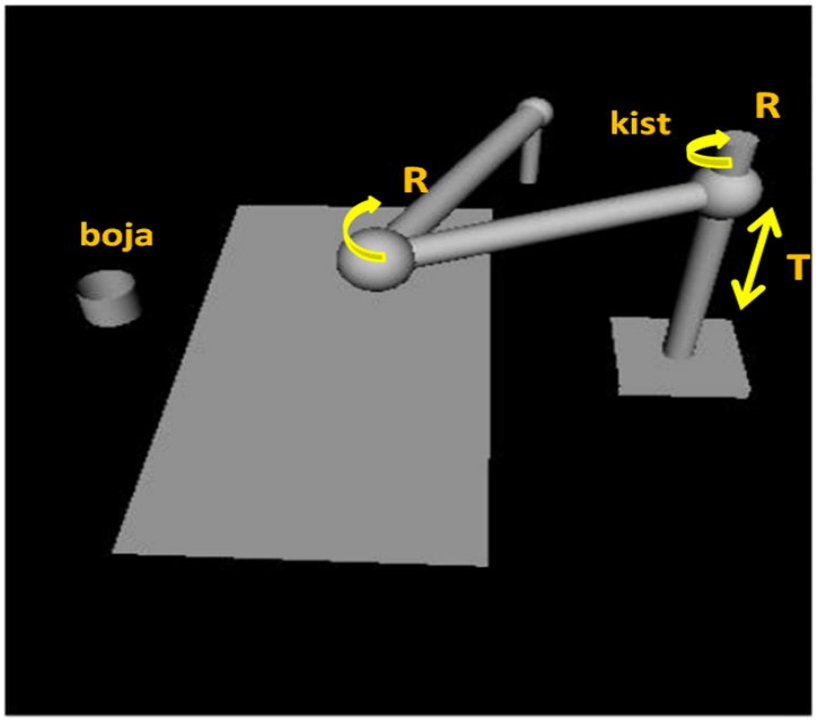
Kako za upravljanje po zglobovima (opcija grafičkog sučelja) nije bila potrebna inverzna kinematika, slanje vrijednosti u tom načinu rada je izvedeno tako da se preko sklopke omogućuje direktno slanje kuta u VR model.

2.4. Virtualni model

Prije pokušaja manipuliranja virtualnim modelom napravljen je detaljan model robotske ruke od LEGO dijelova u 3Ds Max programu, no pokušaj manipuliranja takvim modelom bio je neuspješan jer je zbog prevelike složenosti modela simulacija bila prespora. Nakon tog modela napravljen je jednostavan model u 3D World Editoru Matlaba. Tom jednostavnom modelu su duljine članaka i sve ostale dimenzije jednake dimenzijama zadanim u PDF-u "1. seminarski zadatak". U Simulinku je omogućeno upravljanje virtualnim modelom i to rotacijom prvog i drugog članka, te visinom prvog zgloba.



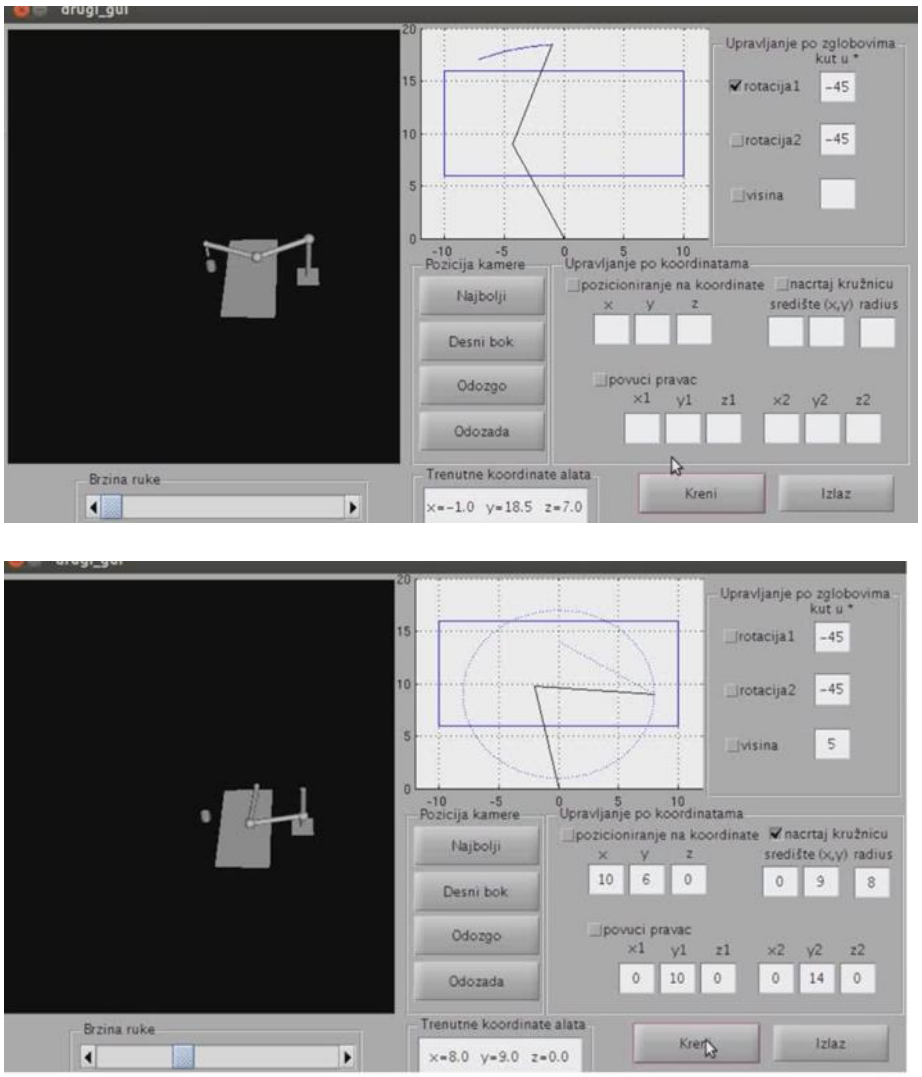
Slika 7. Prvotni izgled virtualne robotske ruke



Slika 8. Pojednostavljeni modele robotske ruke i prostora

2.5. Simulacija

U simulaciji, robotska ruka može napraviti sve potrebne zadatke. Radni prostor robotske ruke je puni cilindar radijusa 20 cm. U model nije isključena mogućnost preklapanja elemenata robotske ruke, odnosno jedan element robotske ruke može proći kroz drugi element.
Kada vrh alata modela robotske ruke prolazi točkom (0,0,z) može doći do skoka od 180 stupnjeva kod oba zgloba. To je posljedica definicije funkcije atan2(y,x) koja se koristi za inverznu kinematiku.
Mijenjanjem kliznog gumba za mijenjanje brzine u simulaciji je moguće osjetiti brže odnosno sporije pokrete. No pri većim brzinama uočava se diskretnost pokreta robotske ruke.



Slika 9. i Slika 10. Simulacija virtualnog modela pomoću grafičkog sučelja

3. Robotska ruka

Za izgradnju robotske ruke korišten je LEGO Mindstorm NXT komplet. Translacijski i rotacijski zglobovi su ostvareni servomotorima i zupčanicima. Svi servomotori su žičano povezani sa NXT kutijom koja prevodi naredbe s računala koje se šalju USB kabelom u naponske signale koji pokreću servomotore.

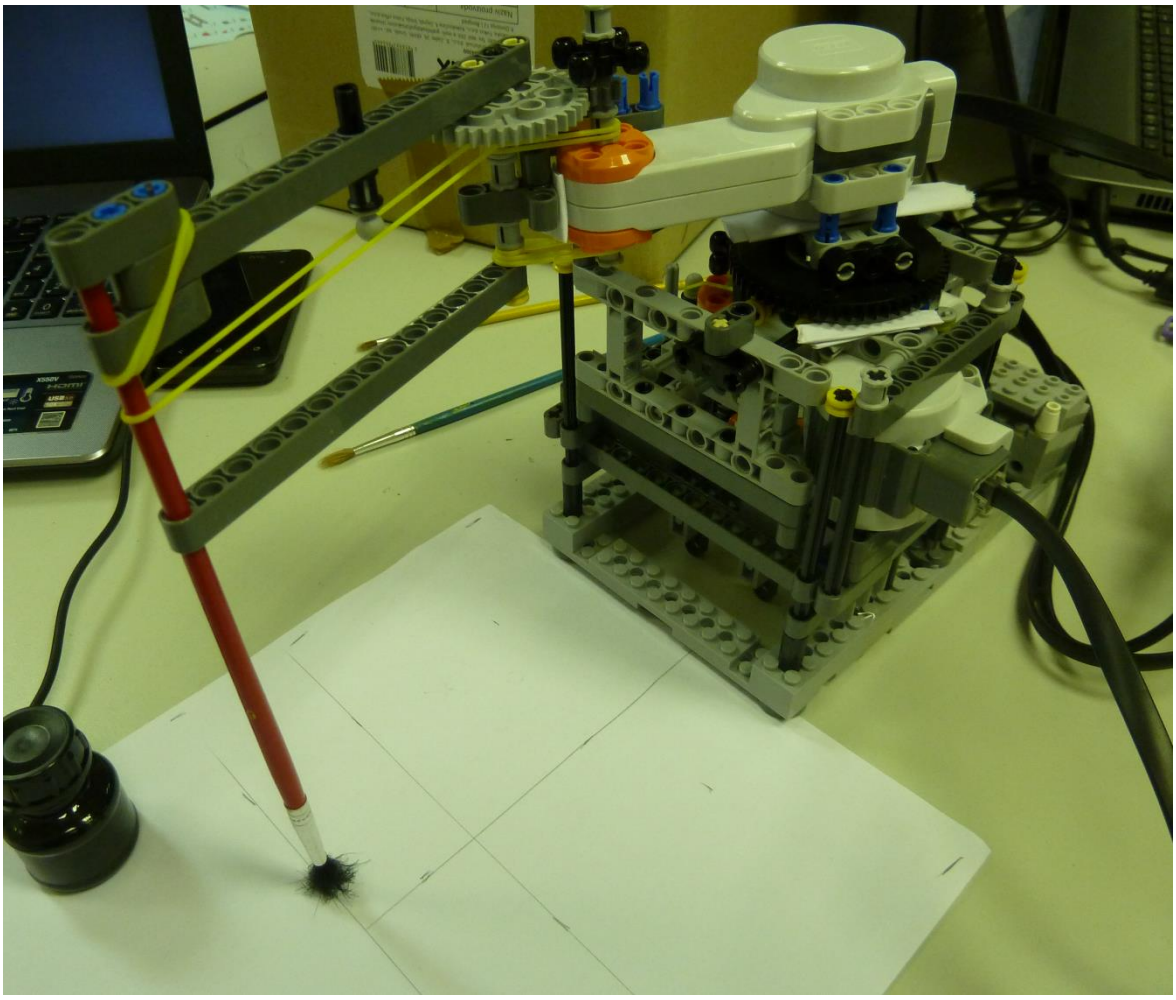
3.1. Arhitektura robotske ruke

Robotska ruka je TRR konfiguracije. Dimenzije robota su promijenjene, jer bi gradnja robotske ruke dimenzija koje se zadane bila znatno teže za rotacijske dijelove, dok bi za translaciju bila nemoguća zbog nedostatka potrebnih dijelova. Bitne dimenzije napravljenog robota su:

- maksimalna visina 4cm
- duljina prvog članka 9.1 cm
- duljina drugog članka 11.3 cm.

Robot ima dovoljno veliki doseg za potrebe izvršavanja danih zadataka.

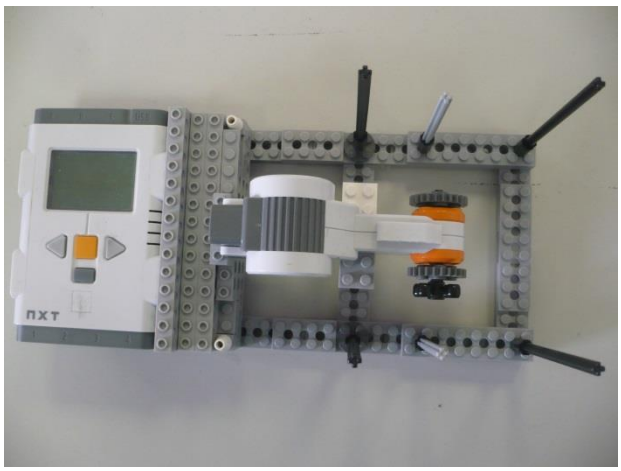
Glavni dijelovi robota su nepomična baza na kojoj se nalaze NXT kutija i servomotor za vertikalnu translaciju, te pomična platforma na kojoj se nalaze druga dva (rotacijska) zgloba.

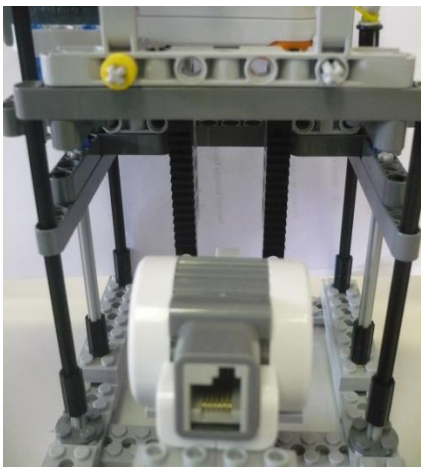


Slika 11. Izgled realizirane robotske ruke

3.1.1. Translacijski zglob

Translacijski zglob se sastoji od servomotora i para ravnih zupčanika koji pretvaraju kružno gibanje motora u pravocrtno gibanje. Translacijom se cijela gornja platforma podiže.



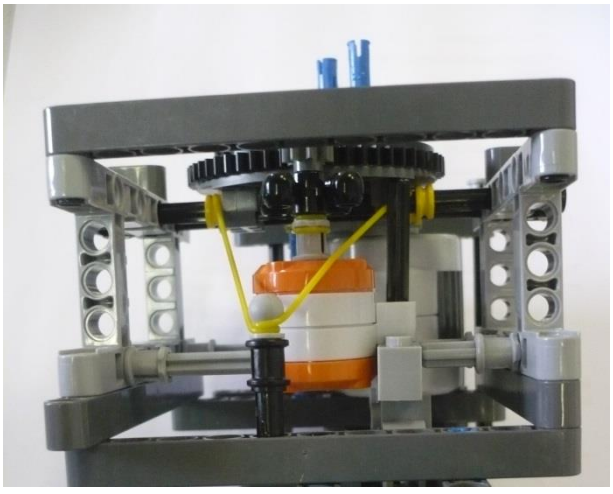
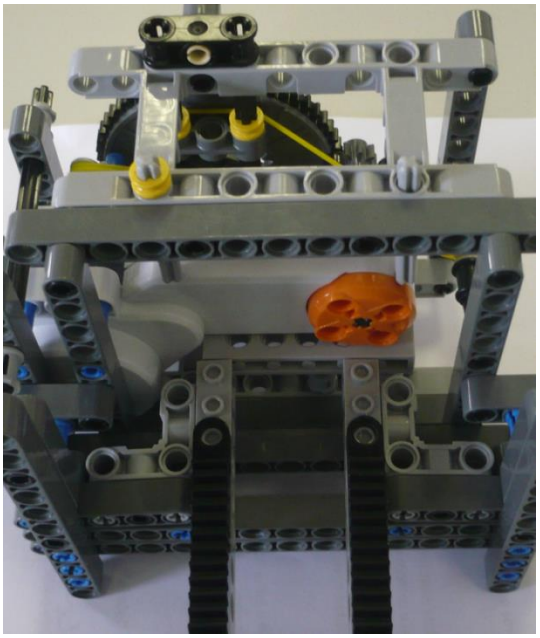


Slika 12. i Slika 13 Izgled translacijskog zgloba

Na slikama 12. i 13. je prikazano kako je ostvarena translacija. Zbog nedostatka ravnih zupčanika robot može promijeniti visinu za 4 cm. Crni i sivi štapići su imali ulogu vodilice koja nije dozvoljavala platformi da se naginje kao i da se previše podigne. Da se platforma podigne za 1 cm na računalo mora poslati naredbu da servomotor zakrene u negativnom smjeru za 48°, i obrnuto, ako se platforma želi spustiti za 1 cm onda se na servomotor mora poslati naredba da se zakrene u pozitivnom smjeru za 48°.

3.1.2. I. Rotacijski zglob

Prvi rotacijski zglob, prikazan na slikama 14. i 15. nalazi se na pomičnoj platformi.

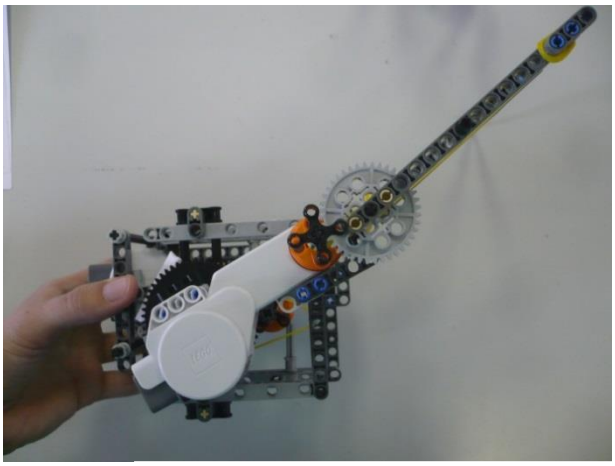
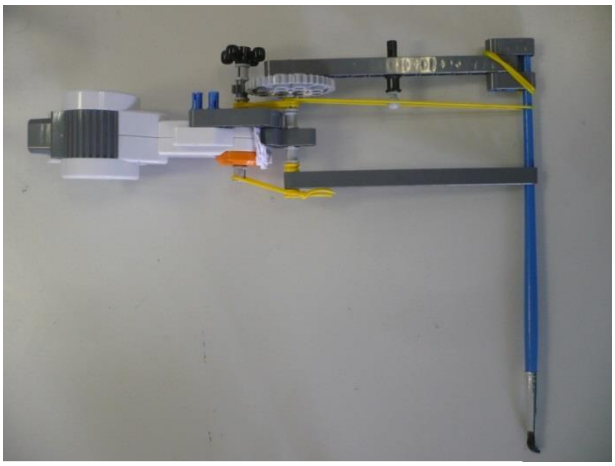



Slika 14. i Slika 15 Izgled I. rotacijskog zgloba

Servomotor prenosi rotaciju na postolje koje ima gornji pomični dio (koji se može rotirati), te donji statični dio koji je pričvršćen na postolje. Napravljen je veliki prijenos (7:1) radi što veće preciznosti, te se servomotor treba okrenuti 7 puta za 360 stupnjeva da se pomični dio postolja okrene jednom za 360 stupnjeva.
Duljina prvog članka je određena duljinom servomotora i zupčanika za prijenos te iznosi 9.1 cm.

3.1.3. II. Rotacijski zglob

Drugi servomotor nalazi se na rotirajućem postolju kojeg zakreće I. rotacijski zglob. Servomotorom te zupčanim prijenosom napravljen je drugi rotacijski zglob, prikazan na slikama 16. i 17.
Omjer prijenosa je 5:1, odnosno servomotor mora napraviti 5x360 stupnjeva da vi se drugi članak okrenuo jednom oko svoje osi.

Slika 16. i Slika 17. Izgled II. rotacijskog zgloba

U produžetku drugog rotirajućeg zgloba nalazi se drugi članak na kojem se nalazi kist. Kako bi se povećala čvrstoća članka i kista, klimavi dijelovi su elastičnim gumicama povezani za čvrste dijelove servomotora. Duljina drugog članka je 11.3 cm.

Zbog velikog kraka koje je imao kist (Slika 17.) prilikom crtanja po papiru mala sila trenja bi uzrokovala dovoljno velik moment da nakrivi kist i iskrivi oblik koji se treba iscrtati.
Također zbog nedovoljno preciznog mjerenja duljina krakova stvarni model ne odgovara točno matematičkom modelu. Kako se kalibracija vrši ručno, a nema povratne veze o stvarnoj poziciji zglobova, malo odstupanje koje postoji na početku tijekom vremena može se akumulirati i uzrokovati i puno veća odstupanja kroz protek vremena.

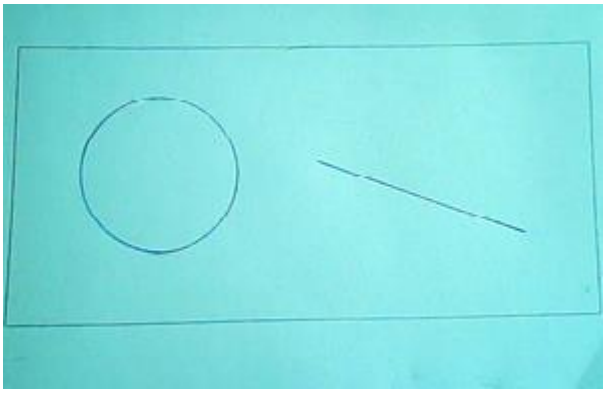
3.2. Programska podrška

Prvi dio, odnosno virtualna simulacija robotske ruke, napravljena je u cijelosti u Matlabu i Simulinku. Za drugi dio zadatka, odnosno upravljanje LEGO NXT robotom, profesori i studenti sveučilišta u Aachenu napravili su funkcije za Matlab kojima se može upravljati pojedinim servomotorima. Postojala je mogućnost upravljanja uz pomoć Simulinka, no kako se u početku upravljanje razvijalo na operacijskom sustavu Linux, za koje nije omogućeno upravljanje u Simulinku, upravljanje robotom je u potpunosti napravljeno u m-funkcijama.

Kako bi NXT kutija prepoznala naredbe iz Matlaba na NXT je prebačen program „MotorControl22.rxe“ također napravljen na sveučilištu u Aachenu.
Napravljenim m-funkcijama moguće je servomotorima zadati kut za koji će se okrenuti, snagu odnosno brzinu kojom će se okretati i dodatne parametre o kojima će ovisnosti brzina i točnost pozicioniranja u konačnu točku.
Ograničenja koja korištene funkcije imaju je moguće zadavanja samo cjelobrojnu vrijednost kuta zakreta te nemogućnost zadavanja nove naredbe dok prethodna naredba nije izvršena, odnosno prekid izvršavanja trenutne naredbe i trenutno izvršavanje nove naredbe.

3.3. Obrada slike

Zadatak obrade slike je bio uzeti sliku pomoću web-kamere i na njoj prepoznati specifične oblike (kružnice i pravci) te kao izlaz dati potrebne podatke za generiranje putanje robotske ruke.
Kamera daje sliku rezolucije 640x480 piksela.



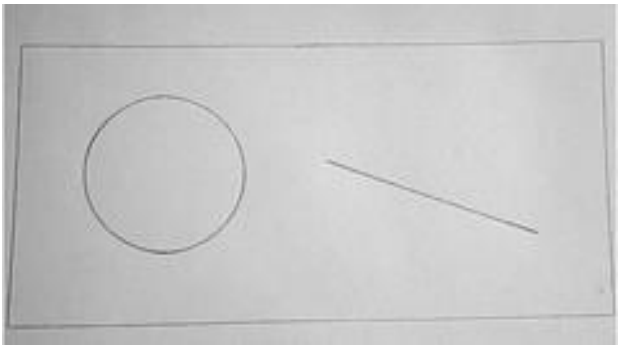
Slika 18. Fotografija dobivena web kamerom

3.3.1. Obrada slike

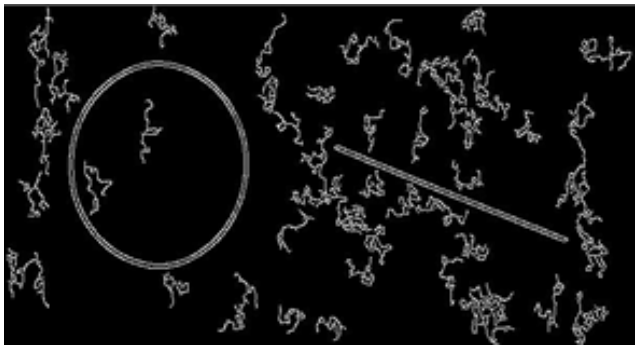
Platno na kojem se nalazi slika koju je potrebno precrtati je dimenzija 20x10cm. Na slici se nalaze jednostavni geometrijski oblici, poput kružnice i pravca, i ta slika se dobiva web-kamerom. Nakon toga algoritmima napisanim u MATLAB-u se slika obrađuje i iz nje se izvlače podaci potrebni za pozicioniranje robotske ruke.

Obrada slike u Matlab-u vrši se na binarnoj slici, no kamerom se dobiva slika u RGB formatu (tri dimenzije podataka tipa uint8). Zato je prvo potrebno sliku prebaciti u binarni oblik. Postoji gotova naredba (3.1) u MATLAB-u za prebacivanje slike iz RGB formata u grayscale format (Slika 19.).
Pretvaranjem slike u sivo područje smanjen je utjecaj sjene na liniju, te se ne javljaju poteškoće pri prepoznavanju linije. Uz to, jednostavnije je i brže koristiti jednu matricu intenziteta boja (sive boje) nego tri matrice intenziteta boja (crvena, zelena i plava).

pict=rgb2gray(pict) (3.1)



Slika 19. Izgled slike u grayscale formatu



Slika 20. Crno-bijela slika sa detektiranim rubovima i šumom

Za detekciju rubova su razvijeni brojni algoritmi, a dijele se na dvije glavne vrste: detekcija promjene intenziteta i Laplaceov pristup. U ovom radu koristi se metoda detekcije promjene intenziteta. Rubovi ustvari predstavljaju gradijent intenziteta na slici, tj. kada dolazi do prijelaza iz tamnoga u svijetlo ili obrnuto.

Kada se odredi matrica gradijenata, potrebno je odvojiti zanimljive objekte od podloge. Najjednostavnije je koristiti binarno ograničenje (engl. tresholding). Binarnim ograničenjem dobiva se binarna slika na kojoj su prepoznati i označeni željeni objekti (3.2).

$$v(x,y) = \begin{cases} 0\,, & za\ u(x,y) \leq T \\ 255\,, & za\ u(x,y) > T \end{cases} \qquad (3.2)$$

Pri tome su: T-prag odluke, u(x,y)- ulazna vrijednost na poziciji (x,y) i v(x,y)- izlazna vrijednost. Relacija se može opisati riječima: u matrici gradijenata svi pikseli, čija je vrijednost manja i jednaka od praga odluke (T), proglašavaju se crnim pikselima (intenziteta 0), a svi ostali pikseli čija je vrijednost veća od praga odluke (T) proglašavaju se bijelim pikselima. Na taj način je dobiven rub objekata.
Pri detekciji rubova korištena je Canny-jeva metoda detekcije, za čiju primjenu također već postoji gotova naredba u Matlab-u. Canny-jeva metoda uzima binarnu sliku (u ovom slučaju u grayscale formatu) i traži rubove na njoj, te vraća prikaz crno-bijele slike sa jasno istaknutim rubovima. Ova naredba (3.3) ujedno prebacuje sliku u binarni format.

```
pict=edge(pict, 'canny')
```
(3.3)

Također je korisno i očistiti sliku od raznih šumova, no ne smije se pretjerati sa „čišćenjem“ slike jer bi se moglo obrisati sa slike i ono što je potrebno. Naredba za čišćenje slike je (3.4). Broj 30 predstavlja primjer zadavanja koliko najmanje piksela mora imati neki rub da ga se ne bi izbrisalo sa slike. Slika prije „čišćenja“ prikazana je na slici 20.

```
pict=bwareaopen(pict, 30)
```
(3.4)

Nakon navedene predobrade slika je spremna za detekciju geometrijskih oblika.

3.3.2. Prepoznavanje linija

Za prepoznavanje linija korištena je Houghova transformacija. Pretpostavka je da su prije transformacije detektirani rubovi. Kombiniranjem rubnih piksela detektiraju se linije, a njihovim kombiniranjem i složenije konture. Nakon detekcije rubova Sobelovim gradijentnim filtrom dobiva se slika rubova koja predstavlja ulazni podatak za Houghovu transformaciju. Radni prostor transformacije je ravnina s dvije vrste piksela: pikselima koji predstavljaju rub i pikselima koji predstavljaju pozadinu. Općenito se kroz svaki piksel ruba može provući beskonačno mnogo pravaca koje možemo prikazati kao:

$$y = a * x + b \qquad (3.5)$$

Slijedeći zadatak je odrediti koeficijente *a* i *b* tako da na tom pravcu leži što više rubnih piksela (ili dovoljan broj što može biti i je jedan od parametara transformacije). Zbog vrijednosti parametara *a* i *b* koji mogu poprimiti sve vrijednosti iz **R**, najčešće korištena transformacija je transformacija pomoću polarnih koordinata. Ako se varijable *x* i *y* zamijene relacijama (3.6) i (3.7) gdje je r udaljenost od ishodišta slike do pravca, a θ kut te okomice iz ishodišta na pravac.

$$x = r * \cos(\theta) \qquad (3.6)$$
$$y = r * \sin(\theta) \qquad (3.7)$$

Slijedi relacija za Houghovu transformaciju u slobodnom prostoru.

$$r = x * \cos(\theta) + y * sin(\theta) \qquad (3.8)$$

Ako se za isti (x ,y) θ mijenja diskretno (1°,2°, ...360°) tada se dobiju različite vrijednosti *r.* Točke *(x,y)* koje leže na istom pravcu, u parametarskom prostoru (θ, *r*) predstavljaju jednu točku. Cilj ovog algoritma je pronaći najdužu liniju na slici, odnosno liniju koju predstavlja najveći broj piksela.

Prije samog algoritma pronalaska linije stvoreno je tzv. akumulatorsko polje Houghove transformacije koje se na početku inicijalizira na nulu (Hough[θ][r]=0;). Za svaki rubni piksel (*x,y*) računaju se vrijednosti za *r,* pri čemu se θ kreće u granicama od (1, 360). Pri tome se *r* kreće u granicama od (0, 420), što je posljedica rezolucije slike. Za svaki rubni piksel (*x, y*) i za svaki θ u granicama od (0, 360 stupnjeva) računa se r iz relacije te se vrijednosti svaki put u akumulatorskom polju (za trenutni *r* i) uvećaju za 1 . Lokalni maksimum unutar akumulatorskog polja određuje zapis pravca:

$$y = -\frac{\cos(\theta_m)}{\sin(\theta_m)}x + \frac{r_m}{\sin(\theta_m)} \qquad (3.9)$$

To je ujedno najduža i najizraženija linija na slici, odnosno linija predstavljena najvećim brojem rubnih (bijelih) piksela. Da bi eliminirali linije koje su manje u odnosu na liniju koju očekujemo, koristimo uvjet za postojanje linije, a to je broj glasova za pojedinu liniju.

Uvjet se svodi na definiranje donje vrijednosti lokalnog maksimuma koji prihvaća liniju kao traženi objekt. U (θ, *r*) prostoru uvijek će postojati barem jedan maksimum. On će definirati najveći pravac unutar slike, ako je njegov broj glasova veći od donje granice lokalnog maksimuma. Sljedeći, lokalni, maksimumi definirat će pravce koji su kraći.

3.3.3. Prepoznavanje kružnica

Nakon predobrade slike, detekcija kružnice vrlo je jednostavna.
Postoji već gotov algoritam za detekciju kružnice i naredba za pokretanje tog algoritma (3.10).

[centar, radijus]=imfindcircles(pict, RADIJUS, 'ImeParametra', 'parametar') (3.10)

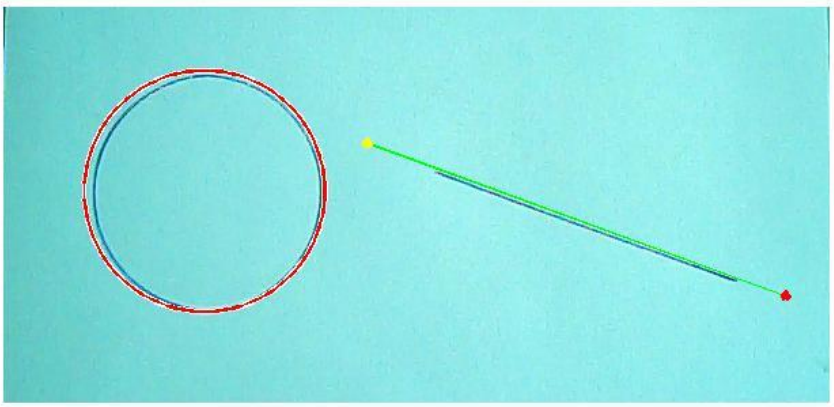
Ova naredba vraća koordinate centara detektiranih kružnica u pikselima i njihove radijuse.
Dodatni parametri koje valja podesiti u naredbi su sljedeći:

- RADIJUS - traženi radijus kružnice, može se zapisati u obliku [Rmin Rmax] pri čemu se traže kružnice radijusa unutar tih granica
- 'ObjectPolarity' - definira kakvi su traženi rubovi u odnosu na pozadinu, u seminaru je postavljeno na 'bright'
- 'Sensitivity' - vrijednost od 0 do 1 kojom određuje kolika će biti osjetljivost detekcije kruga, tj. koliko krug mora biti „savršen“ da bi ga se detektiralo (pritom 0 označava potpuno savršen krug)

Korištena je i funkcija (3.11)

help=viscircles(centar,radijus) (3.11)

pomoću koje se grafički prikažu detektirani krugovi (crvenom bojom).



Slika 21. Prikaz detektiranog kruga i linije

3.3.4. Kalibracija slike

Slika koja je dobivena web-kamerom sadrži sliku sa geometrijskim likovima i okolni prostor, no dimenzije dobivene slike nisu jednake pravim dimenzijama. Zbog toga je potrebno izvršiti kalibraciju slike.
Kako osim željene slike imamo i okolni prostor na snimljenoj slici, moramo iz ukupne slike uzeti samo željeni dio.Naredba za izrezivanje dana je (3.12).

i=imcrop(pict, [x1, y1, x2, y2] (3.12)

gdje su zadane koordinate sporedne dijagonale pravokutnika.
Nakon što smo dobili željenu sliku, provodi se obrada slike opisana u prethodnim poglavljima.
Nakon obrade slike potrebno je robotu zadati koordinate u centimetrima u odnosu na neki proizvoljno određeni koordinatni sustav. Kako koordinate bile u centimetrima potrebno je preračunati koliko je piksela u jednom centimetru, korištene funkcije za prepoznavanje vraćaju koordinate kružnica i pravaca u pikselima. Budući da su dimenzije slike u pikselima poznate, te je također poznato koliko je to u stvarnosti centimetara, moguće je izvršiti skaliranje iz omjera piksela i centimetara.

3.4. Upravljanje robotom

Za upravljanje robotom korištene su iste funkcije kao i za virtualni model. Pokretanje samog virtualnog modela istovremeno sa upravljanjem pravom robotskom rukom je maknuto jer je istovremena simulacija usporavala izvršavanje programa.
Funkcije šalju na svaki pojedini motor kut za koji se servomotor treba zakrenuti te se program za upravljanje robotskom rukom napravljen za virtualni model morao malo izmijeniti. U svakom koraku se pamtio trenutni kut, te bi se izračunao kut u kojem se zglob trebao nalaziti na kraju tog prolaza petlje. Razlika ta dva kuta bi se poslala na servomotor. Kako su postojala tri neovisna servomotora naredbe bi se poslale na sva tri motora te bi se zatim pozvala funkcija „WaitFor()“ za svaki motor. Time bi program čekao da se izvrši gibanje sva tri motora prije nego što mu pošalje nove naredbe.
Funkcija „kutevi.m“ je uz postojeće argumente, koji su bili kut prvog članka u odnosu na mirujući sustav, te kut drugog članka u odnosu na mirujući sustav, vraćala i razliku ta dva koja te je ta vrijednost imala značenje kuta za koji se drugi zglob treba pomaknuti u pomičnom sustavu prvog zgloba.

Kod pozicioniranja u određenu točku nije bila bitna trajektorija te se servomotorima mogao poslati ukupan kut za koji se trebaju okrenuti da se vrh alata nađe u željenoj točki. Zbog toga se pozicioniranje izvršavalo kontinuirano.
S druge strane kod crtanja kružnice ili linije bilo je potrebno paziti da se u svakom trenutku imamo točno određenu kombinaciju kutova. To se postiglo slanjem potrebnog pomaka kutova tako da alat dođe iz jedne točke u drugu. Kako se za slanje nove naredbe na motor mora provjeravati da li je gotova prethodna naredba gibanje robotske ruke prilikom iscrtavanja linija i kružnica je diskretno.

Zbog mogućnosti slanja samo cjelobrojnih kutova servomotoru nije se mogla postići tako velika preciznost kao u matematičkom modelu. Zbog toga napravljen spremnik u koji se spremao višak odnosno manjak stvarnog kuta i najbližeg cjelobrojnog kuta po formuli (4.1).

$$Spremnik = Spremnik + kut - round(kut) \qquad (4.1)$$

Kada bi se kut slao na servomotoru, trenutnom kutu bi se poslala i razlika kuteva od prije, odnosno:

$$posalji\_motoru(kut + spremnik) \qquad (4.2)$$

Te bi se na kraju prolaska petlje u spremnik ponovo pospremila razlika postojećeg i poslanog kuta:

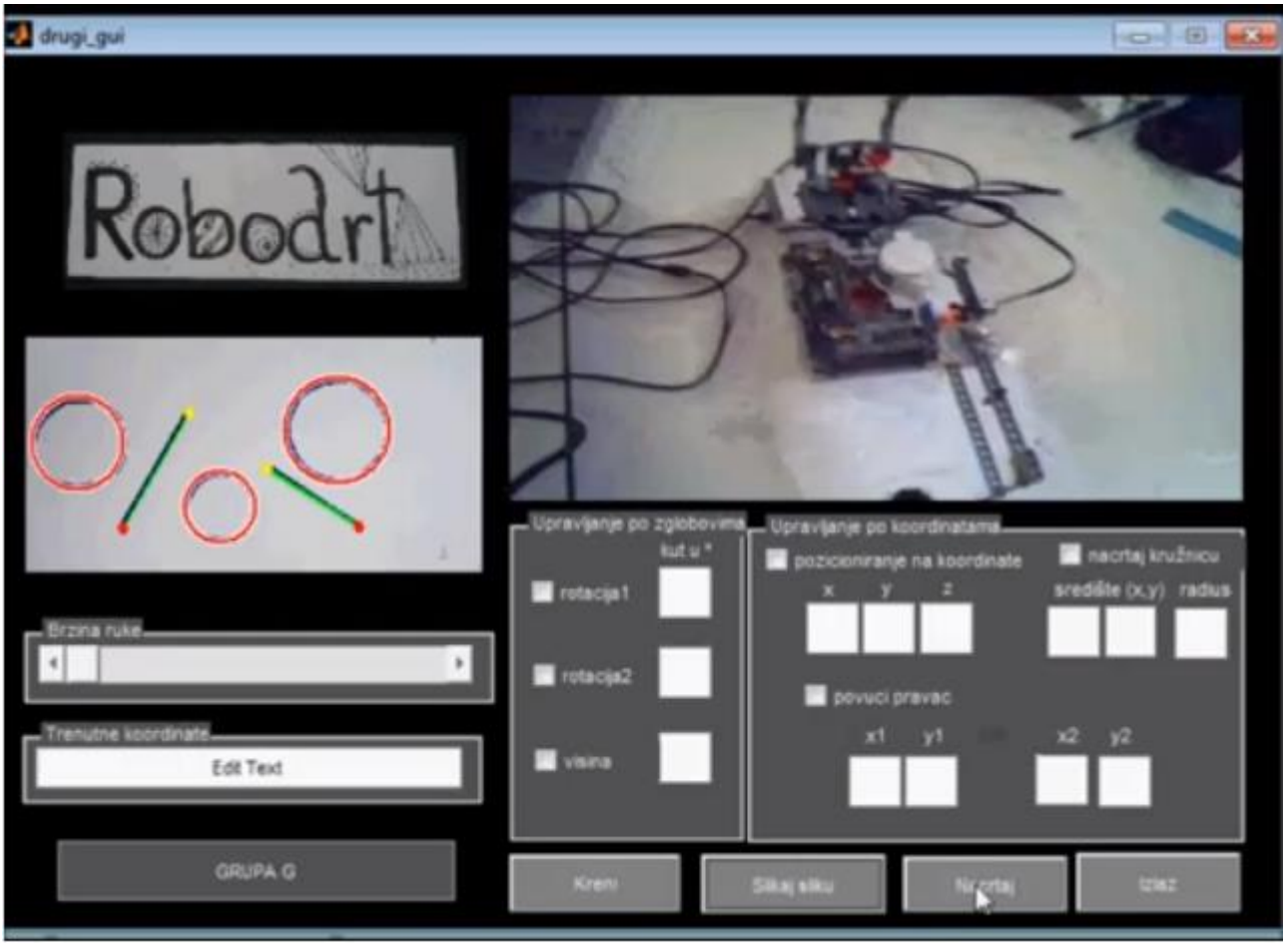
$$Spremnik = kut + Spremnik - round(kut + Spremnik) \qquad (4.3)$$

Ovim postupkom je smanjena greška prilikom crtanja kružnice i linije.

Robot nema automatsku kalibraciju, već je na početku upravljanja potrebno namjestiti robotsku ruku u određen položaj tako da se taj položaj slaže za pretpostavljenim početnim položajem u programu. Položaji zglobova robotske ruke se pamte u programu, no zbog razlike matematičkog i stvarnog modela, nakon određenog vremena položaji u računalu i stvarni položaji zglobova mogu se podosta razlikovati.

3.5. Grafičko sučelje

Grafičko sučelje za upravljanje robotskom rukom malo je izmijenjeno u odnosu na grafičko sučelje za upravljanje virtualnim modelom (Slika 22.) Glavna razlika je što, umjesto virtualnog modela i grafa na kojem su pamćene prijeđene točke robota, novo sučelje ima dio u kojem je slika kamere koja snima crtež za precrtavanje, te dio u kojem je slika kamere koja snima robotsku ruku dok crta. Zbog nepotrebnosti crtanja pravca u 3D prostoru, maknute su z- koordinate kod crtanja pravca.

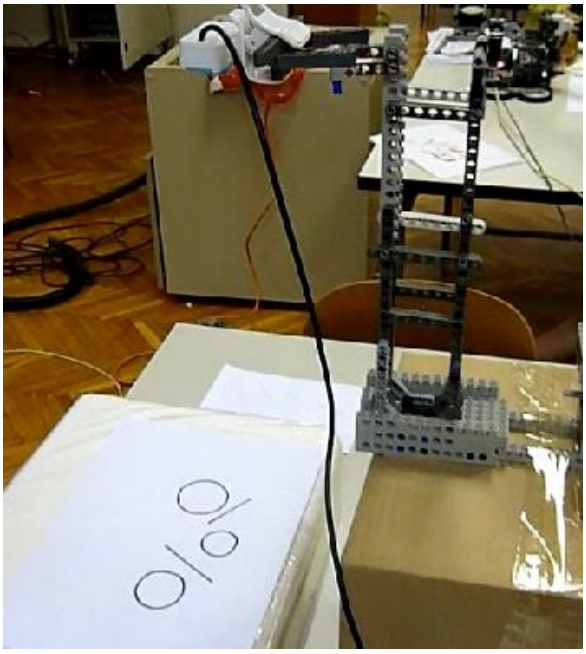


Slika 22. Upravljanje robotskom rukom pomoću grafičkog sučelja

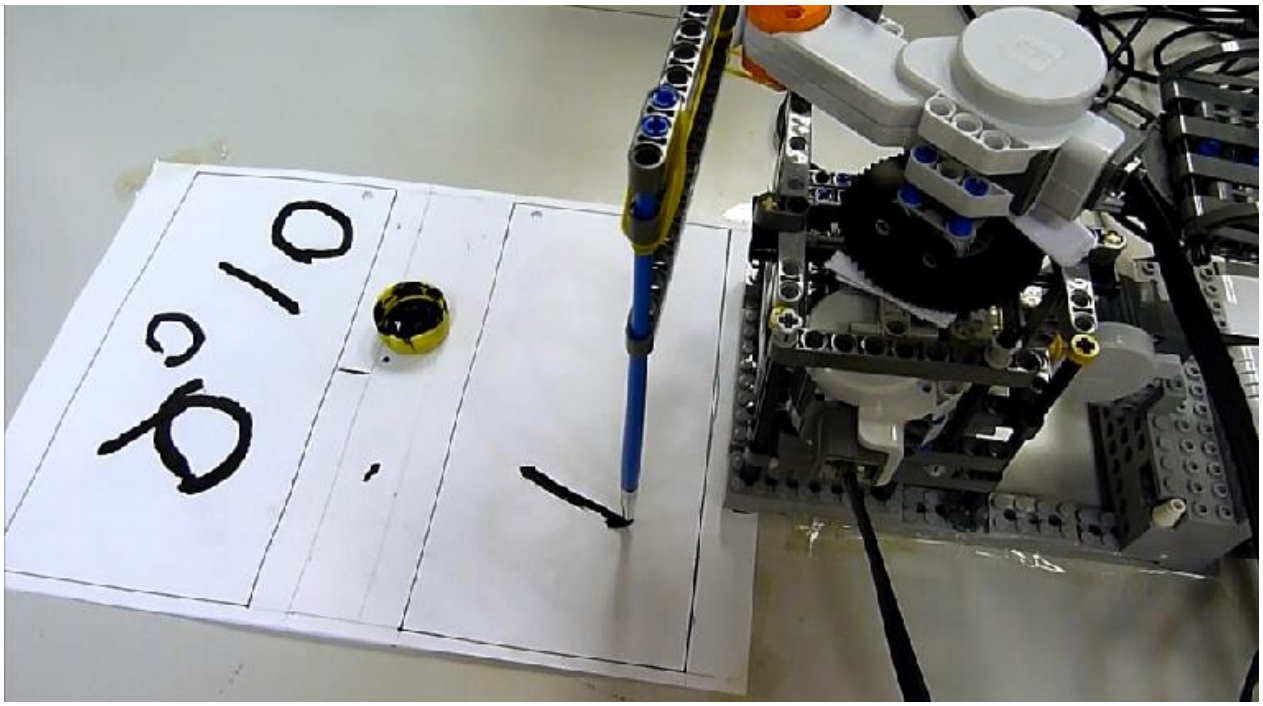
Uz postojeće gumbe dodan je gumb „Slikaj sliku“ čijim će se pritiskom uslikati slika (slika lijevo) te će se programom za obradu slike označiti kružnice i linije. Nakon što je slika obrađena pritiskom gumba „Nacrtaj“ robotska ruka će krenuti crtati prepoznate linije i kružnice.

4. Zaključak

Na Slici 23. prikazan je primjer zadane slike, te web kamera koja se nalazi iznad slike.
Na Slici 24. prikazan je robot prilikom crtanja zadane slike. Također, na papiru se nalazi već nacrtana ista slika.



Slika 23. Zadana slika za precrtavanje te web kamera iznad nje



Slika 24. Robot za vrijeme precrtavanja slike te već precrtana slika

Izvršen je zadani zadatak; robot je sposoban precrtati zadanu sliku, te je omogućeno upravljanje robotom upravljajući svakim zglobom pojedinačno. Robot nije robustan, jer mu se zbog nepostojanja povratne veze greška u pozicioniranju nakuplja kroz vrijeme. Također zbog

mogućnosti slanja samo cjelobrojnih kutova zakreta, pomicanje sa malom promjenom kuta je neprecizno. Zbog velikih dimenzija krakova i kista, trenje kista sa papirom stvara veliki moment na robota koji motor ne može savladati i stoga su ponekad likovi malo iskrivljeni.
Ovi navedeni nedostatci mogli bi se doraditi, no potrebno je puno mašte i dosta vremena.

Ipak, ovaj seminar donio je mnogo novih praktičnih znanja, kao i zanimljivih i zabavnih trenutaka.