Univerza v Ljubljani
Fakulteta za elektrotehniko in računalništvo
Laboratorij za robotiko
Tržaška 50, 61001 Ljubljana

Poročilo z laboratorijskih vaj pri predmetu Osnove robotizacije

Leon Kos, sodelavca: Simon Mali in Goran Nagy

Štud. leto: 1993/94

1 Seznanjanje z robotom Seiko RT-2000

1.1 Naloga

Z robotom SEIKO RT-200 realizirajte "Pick & Place" rutino in jo preizkusite v STEP, WALK in INTERPRETER načinu delovanja programa.

1. HOME procedura, gibanje segmentov s pomočjo tipk pri razli"nih hitrostih, JOGSP(1-50)
2. definiranje točk s pomočjo ukaza HERE
3. testiranje DISPLAY načina delovanja (odčitavanja koordinat, definiranje točk)
4. gibanje robota z ukazom MOVE
5. EDIT način delovanja in "PICK & PLACE" rutina
6. izvajanje programa v STEP, WALK in INTERPRETER način delovanja
7. gibanje robota pri različnih programskih hitro0stih SPEED
8. izvajanje programa pri ukazih SHAVE, NOSHAVE, SYNC, NOSYNC

Slika 1: Robot SEIKO RT-2000

1.2 Potek vaje

Robot SEIKO RT-2000 je robot cilindričnega tipa, ki ima za svoje upravljanje posebno konzolo z LCD zaslonom. S to konzolo nadzorujemo delovanje robota in pisemo programe v jeziku, ki je BASIC z razširitvami ukazov za robota (DARL).

Konzola ima več načinov delovanja (MODE). Ti načini so predstavljeni s prvo črko v zgornjem levem kotu LCD zaslona:

HOME inicializira položaj robota v izhodiščno lego in opravi potrebne inicializacije servo krmilja

WALK omogoča spremljanje programa po ukazih za gibanje.

STEP deluje podobno kot walk, le da tu lahko spremljamo izvajanje programa po programskih korakih, kar je bolj podrobno kot spremljanje premikov robota.

DISPLAY nam sprotno prikazuje stanje robota (hitrost, pozicija, ...)

EDIT v tem načinu vnašamo in modificiramo program, ki ga robot interpretira. V konzolo je vgrajen primitiven editor, ki omogoča osnovne funkcije pisanja programa.

CALIBRATE se redko uporablja, za osnovno nastavljanje natan"nosti robota

Programski jezik vsebuje standardne konstrukte v BASIC jeziku (FOR/NEXT, GOTO, GOSUB / RETURN, IF / THEN / ELSE, STOP). Spremenljivke so samo celoštevilčne (integer) označene z eno samo črko. Točke so posebna vrsta spremenljivk, ki imajo poseben pomen, zato se začenjajo s črko T, kateri sledi številka v mejah

$$T0..T399 = x[mm] \quad y[mm] \quad z[mm] \quad \alpha[\mathrm{deg}]$$

V monitorskem načinu lahko definiramo točke, s prefiksnim ukazom **DO**. Pri ročnem pomikanju robota se lahko nastavlja hitrost z ukazom JOGSP (Jog speed). Tako smo npr. definirali točke z ukazom:

```
DO HERE T1
DO HERE T2
...
```

Prijemalo na robotu je bilo priklopljeno na izhodno linijo 0 in 1, tako se prijem elementov kontrolira z ukazom

```
DO OUTPUT +OG0 -OG1
DO OUTPUT -OG0 +OG1
```

Pomik na določeno točko se izvede z ukazom MOVE, ki pa gre lahko preko večih točk. Kako se giblje skozi vmesne točke se kontrolira z ukazom SHAVE ali NOSHAVE. Pri NOSHAVE bo sel robot zanesljivo skozi vmesno točko, sicer pa bo pot optimiral glede na ojačanje servo krmilnika in trenutne hitrosti.

Ker ima robot tudi možnos rotacije prijetega elementa je mogoče tudi sinhrono vrtenje predmeta skupaj z premikanjem roke robota. Na ta način ostane predmet vedno orientiran v eno smer. To kontroliramo z ukazom SYNC/NOSYNC.

1.3 Program

Za zahtevano "pick & place"rutino sme napisali program z uporabo podprogramov, ki so skrajšali in poenostavili praogram. Tako je na vrsticah:

200 prihod v točko

300 odhod iz točke

```
LLIST 10-400
 00010 I=1
 00015 GOSUB 200
 00020 GOSUB 300
 00030 I=I+1
 00040 GOSUB 200
 00050 IF I < 5 THEN GOTO 20
 00054 I=I-1
 00055 GOSUB 300
 00060 I=1
 00070 GOSUB 200
 00075 GOTO 10
 00080 END
 00200 MOVE T(I+10)
 00205 NOSHAVE
 00206 SPEED 50
 00210 MOVE T(I)
 00220 OUTPUT -OG0 +OG1
 00230 DELAY 500
 00240 RETURN
 00300 OUTPUT +OG0 -OG1
 00320 DELAY 500
 00325 SHAVE
 00326 SPEED 260
 00330 MOVE T(I+10)
 00340 RETURN
 00400 END
E> TLLIST
T0=      407.999 -166.413  -45.156  -22.189
T1=      472.436 -203.94   -97.444   90.269
T2=      518.701   26.872  -94.828   93.208
T3=      334.966   81.937  -94.828  149.721
T4=      315.494 -136.192  -97.428   90.269
T5=      422.986  -71.127  -96.597   90.269
T6=      492.355  131.804  -39.841  -93.696
T7=      277.1   -239.999  -92.678  -52.874
T8=      519.571 -134.658  -96.734  -26.508
T9=      519.571 -134.658  -43.781  -26.508
T10=     434.357 -187.503  -97.444  -23.349
T11=     472.436 -203.94   -28.472   90.269
T12=     518.701   26.872  -34.297   93.208
T13=     334.966   81.937  -31.113  149.721
T14=     315.494 -136.192  -26.903   90.269
T15=     422.986  -71.127  -44.281   90.269
```

2 Vključevanje vhodnih linij pri SEIKO RT-2000

2.1 Naloga

Definirajte paleto šetih elementov in testirajte digitalno vhodno linijo IE1:

1. s "PICK & PLACE"rutino zgradite piramido
2. s prvimi tremi elementi zgradite stolpec

ali: Definirajte dve paleti šestih elementov in testirajte digitalno vhodno linijo IE1:

1. s "PICK & PLACE"rutine prenesite elemente iz ene palete v drugo
2. druga paleta naj bo sestavljena v obratnem vrstnem redu

ali: Definirajte nov koordinatni sistem in prenesite elemente iz palete v osnovnem koordinatnem sistemu v isto paleto v novem koordinatnem sistemu.

2.2 Potek vaje

Odločili smo se za prvo varianto, ki je nekako najzanimivejša saj ne deluje samo v ravnini ampak pri piramidi zahteva tudi odmik v višino. Operacije s točkami so možne samo tako, da prištevamo ali odštevamo točko, ki jo definiramo popreje. Tako smo izmerili in definirali naslednje točke, ki se uporabijo za relativne premike. Te točke so bile:

T100 pomik za 30 mm navzgor

T200 pomik za 7 mm navzgor

T300 začetek T4

T210 pomik za 10 mm v desno

Slika 2: Sestavljanje piramide

Tako kot v prejšnji vaji smo tudi tu uporabili modularno programje z podprogrami na sledečih vrsticah:

500 pomik nad točko T300

600 spust na točko T300 in povečanje pozicije

Velikost krogcev, ki smo jih prenašali je bila 15 mm v premera in 7 mm višine.

2.3 Program

```
E>LLIST
 00001 T100=0. 0. 30. 0.
 00002 T200=0. 0. 7.01 0.
 00003 T300=T4
 00004 T210=10. -1.5 0. 0.
 00010 DEF PA1(2,1) T1 T2 T3
 00020 IF IE1=1 THEN GOTO 1000
 00030 J=0
 00040 FOR I=0 TO 2
 00050 GOSUB 400
 00060 GOSUB 300
 00070 GOSUB 400
 00080 GOSUB 500
 00090 GOSUB 600
 00100 GOSUB 500
 00120 NEXT I
 00130 END
 00300 MOVE PA1(I,J)
 00310 OUTPUT +OG0 -OG1
 00320 DELAY 500
 00330 RETURN
 00400 T399=PA1(I,J)
 00410 T399=T399+T100
 00420 MOVE T399
 00430 RETURN
 00500 T399=T300+T100
 00510 MOVE T399
 00520 RETURN
 00600 MOVE T300
 00610 OUTPUT -OG0 +OG1
 00620 DELAY 500
 00630 T300=T300+T200
 00640 RETURN
 00700 T10=T10+T100
 00710 MOVE T10
 00720 T10=T10-T100
 00725 MOVE T10
 00730 OUTPUT -OG0 +OG1
 00740 DELAY 300
 00744 T10=T10+T100
 00746  MOVE T10
 00748 T10=T10-T100
 00750 T10=T10+T210+T210
 00760 RETURN
 01000 T10=T5
 01005 K=1
 01010 FOR J=0 TO 1
 01015 FOR I=0 TO 2
 01020 IF K=4 THEN T10=T5+T200+T210
 01030 IF K=6 THEN T10=T5+T200+T200+T210+T210
 01050 GOSUB  400
 01060 GOSUB 300
 01070 GOSUB 400
 01080 GOSUB 700
 01090 K=K+1
 01100 NEXT I
 01110 NEXT J
 01120 END
E>TLLIST
T0=      407.999 -166.413  -45.156  -22.189
T1=      475.348 -216.879  -97.063   -0.003
T2=      481.958 -133.589  -97.063    0.
T3=      435.673 -211.225  -97.063   -0.003
T4=      511.314   53.399  -94.894    0.001
T5=      318.532  158.677  -94.153   -0.004
T6=      522.753  121.719  -85.841    6.838
T7=      522.753  121.719  -79.309    6.838
T8=      510.968   37.46   -77.503   -1.617
T9=      510.968   37.46   -71.075   -1.617
T10=     358.532  152.677  -80.133   -0.004
T11=     301.794  145.246  -95.15    -1.153
T12=     323.452  144.022  -95.15    -1.153
T13=     344.308  145.508  -95.15    -1.153
T14=     334.352  145.47   -88.625   -1.156
T15=     312.461  144.233  -88.625   -1.155
T16=     325.024  145.358  -81.422   -1.155
T20=     383.763  -54.877  -51.884  -16.88
T30=     417.302  -28.617  -24.906  -16.72
T40=     356.885 -258.437  -52.434   -1.675
T41=     424.193  119.233  -52.434    1.426
T100=      0.       0.      30.       0.
T200=      0.       0.       7.01     0.
T210=     10.      -1.5      0.       0.
T251=    510.968   37.46   -73.041   -1.617
T252=    506.486  -77.224  -45.616   -1.621
T253=    490.922 -146.586  -84.875   -1.619
T300=    511.314   53.399  -94.894    0.001
T399=    442.283 -127.935  -67.063   -0.003
```

3 Osnovni programi za robot Accusembler SSR-X

3.1 Naloga

Z robotom Accusembler SSR-X realizirajte Pick-And-Place rutino (primi, prinesi, odloži) in jo izvedite z ukazom RUN ter ukkazom STEP n.

1. Izvedite HOME proceduro
2. Preverite pomikanje segmentov s pomočjo tipk pri različnih hitrostih in različnih korakih.
3. Definirajte točke v prostoru (tipka P* v teach načinu)
4. Preverite stanja na vhodu (tipka SW/E¿ v teach načinu). Z ukazom VER preverite nastavitve parametrov robota.
5. Gibanje robota s pomočjo ukaza MOVE, Go, ARC, CMOVE in CARC.
6. Gibanje posameznih segmentov in testiranje maksimalnih hitrosti.
7. Napišite Pick-And-Place rutino
8. Izvedite to rutino z ukazom RUN in STEP.

3.2 Potek vaje

Tako kot vsi roboti na vajah ima tudi ta vrsta robota vgrajen neke vrsta BASIC jezik z razširitvami za gibanje robota. Tako tu najdemo ukaz PALET za nastavljanje palet v globalnem koordinatnem sistemu. Z ukazom FEED nastavljamo hitrost gibanja robota.

Preverili smo gibanje robota z ukazom CMOVE in CARC, kjer se eje izkazalo, da ta dva ukaza delujeta samo v eni ravnini, medtem ko MOVE in ARC delujeta v vseh dimenzijah. Razlika med ukazi je le ta, da se slednja dva ustavljata v vmesnih točkah. Poleg klasičkih ukazov se

Slika 3: Robot Accusembler SSR-X

izkaže za priročnega tudi ukaz JUMP, ki izvaja skokovite pomike na željene točke z maksimalnim odnikom po osi z. Čakanje za prijem se izvede z ukazom WAIT, ki ima enote v sekundah.

Na izhodno linijo 15 je bil proklopljen vakumski sesalnik, s katerim smo prijemali in spuščali SMD elemente. Vlkop in izklop sesealnika se izvede z ukazom ON / OFF.

3.3 Program

Tabela 1: Listing programa

4 Vključevanje vhodnih linij pri robotu Accusembler SSR-X

1. Definirajte dve paleti šestih elementov in testirajte digitalno vhodno linijo 1
   - s pomočjo Pick-And-Place rutine pinesite elemente iz ene palete v drugo
   - druga paleta naj bo sestavljena v obratnem vrstnem redu
2. ali/ Definirajte nov koordinatni sistem z ukazom LOCAL in prenesite elemente iz palete v osnovnem koordinatnem sistemu v izto paleto v novem koordinatnem sistemu
3. ali / Prenesite elemente iz ene palete v drugo po vnaprej predpisani trajektoriji. Testirajte vhodno linijo 1:
   - za premikanje robota uporabljajte ukaz Move, spreminjajte hitrost gibanja in natančnost interpolacije
   - za premikanje uporabljajte ukaz GO

4.1 Izvedba vaje

Odločili smo se za prvo varianto. Ker ni bili točno specificirano, kaj pomeni ovratni vrstni red, smo si sami izbrali vrstni red, ki je imel nekako zamenjan vrstni red pobiranja in sestavljanja nove palete.

Slika 4: SSR-X palete elementov

V nalogi se zahteva kontrola vhodne linije in odločitev sestavljanja dveh različnih palet glede na stanje vhodne linije, katero se preklaplja s stikalom. Testiranje linije se enostavno izvede z ukazom IF in pa vgrajeno funkcijo SW(). Za pominkanje na točke smo uporabili ukaz JUMP, ki se je izkazal za najugodnejšega.

4.2 Program

Potrebno je bilo definirati tri palete, od katerih je bila prva paleta izvorna (od tam je program pobiral SMD elemente) ostali dve pa sta bili definirani za postaviten v istem in obratnem vrstnem redu.

Tabela 2: Listing programa

5 Seznanjanje z robotom Adept One

5.1 Naloga

Z robotom AdeptOne realizirajte "pick and place" rutino (primi, prinesi, odloži) in jo izvajajte z ukazom XSTEP in EXECUTE. Primerjajte Razliko med MOVE in MOVES.

1. Izvajannje CALIBRATE procedure v monitorju in z ročne naprave
2. Gibanje segmentov robota v *joint state, world state, tool state, free state* načinih z ročno napravo
3. Definiranje točk v monitorju z ukazi *where, point, here, teach*
4. Uporaba EDIT, napiši "pick and place" proceduro
5. Izvajanje programa z XSTEP, EXECUTE
6. Izvajanje programa s PCEXECUTE, TRACE

5.2 Izvedba vaje

Namen vaje je seznaniti se z osnovnimi ukazi in delovanjem robota AdeptOne. Na začetku dela z robotom je treba izvesti ukaz Calibrate z ročne naprave ali iz monitorja. S tem ukazom robot določi svojo začetno pozicijo v prostoru. Ko je robot izvršil ta ukaz lahko pristopimo k programiranju.

Robot ima možnost premikanja v različnih načinih. Prvi način je *World state*. V tem načinu se robot premika z ozirom na svoj osnovni koordinatni sistem. Koordinatni sistem je vedno isti in se s premikanjem robota ne spreminja. Pri premikanju v *Tool state* načinu se robot premika z ozirom na koordinatni sistem orodja. Skupaj s premikanjem robota, se premika tudi koordinatni sistem orodja. V *Joint state* načinu lahko premikamo posamezne sklepe. V *Freee state* načinu lahko sprostimo željene sklepe in tako ročno nastavljamo pozicijo sklepov. Točke lahko definiramo na različne načine. Zajenostavnejši način je, da v enem od načinov premikanja postavimo vrh robota na željeno mesto in ukazom *here* definiramo točko.

Primer: `do HERE t1`

Drugi način je *teach*. V tem načinu definiramo polje točk samo z določitvijo imena. Je bolj primeren za učenje sekvence točk, tako da premikamo robota in pritiskamo na tipko `REC/DONE`.

Razlaga ukazov:

Point izpiše pozicijo in orientacijo vrha robota, lahko pa jo tudi spremenimo.

Where izpiše koordinate točke

Xstep izvrši del programa in se ustavi ko izvede prvi naslednji pomik.

Execute izvrši celoten program

Pcexecute se uporablja za izvajanje večih programov simultano na večih procesorjih, ki jih ima robot AdeptOne.

Move premakne robota v novo pozicijo (točko)

Moves se podobno premakne, vendar gre tu za točno premikanje po premici iz točke v točko

Trace pomaga slediti izvajanje programa. Izpisuje trenutno vrstico programa.

5.3 Program

Prvi program, ki smo ga izvedli se je enostavno premikal iz točke v točko.

```
.PROGRAM prvi()
       SPEED 30 ALWAYS
       MOVE t1
       MOVE t2
       MOVE t3
       MOVE t4
       MOVE t5
       MOVE t6
       MOVE t7
.END
```

Sledil pa je boljši program, kjer smo uporabili *teach* način z večjim število točk.

```
.PROGRAM tretji()
       ACCEL 50, 50
       SPEED 50 ALWAYS
```

```
        FOR i = 1 TO 16
            MOVES tocke[i]
            DELAY 0.1
        END
.END
```

6 Osnovni programi za robot Adept One

6.1 Naloga

Definirajte dve matriki sedmih elementov in elemente iz prve matrike razporedite v drugo. Uporabite "pick and place" rutino

Namen te vaje je bil napisati program, ki bo prenesel 7 elementov iz en palete v drugo. Ker robot AdeptOne nima ukaza podobnega ukazu PALET pri robotu Accusembler, je bilo potrebno vse premike definirati relativno proti prvemu elementu v paleti. Razmak med dvema elementoma, ki ju je bilo treba prenesti je bil 55.5 mm.

Slika 5: Prenos palete elementov

Da bi robot prenesel prvi predmet mora priti v točko P_1 in ga dvigniti v drugo paleto. Točko P_2 dobimo tako, da y koordinati točke P_1 prištejemo 55.5. To naredimo z ukazom:

```
SET P2 = SHIFT(P1 BY 0, 55.5, 0)
```

x in z koordinati točke P_2 sta enaki točki P_1

ukaz TIMER se uporablja za razne namene, v našem primeru smo uporabili za zakasnitev gibov.

6.2 Program

Programski jezik v katerem je program napisan Basicu z nekaterimi spremembami, ter dodanimi robotskimi ukazi.

Program deluje na naslednji način:

Prvo se vrh robota nastavi v luknjo elenenta, ki ga trenutno prestavljamo, tu počaka nekaj časa, ter dvigne prednet in ga položi na pravo mesto v drugi paleti; počaka nekaj časa, nato pa gre na naslednji element.

Za nastavitev hitrosti smo uporabili ukaz:

```
SPEED 40 ALWAYS
```

Atribut always pomeni, da nastavljena hitrost velja za celoten program in ne samo za naslednji MOVE ukaz.

Poleg naštetih ukazov so tu še klasični robotski ukazi (move, ...) ter zanka, katere sintaksa je podobna Basicu, le da zanko zaključimo namesto z NEXT z END, kot tudi sam ukaz.

```
.listp paleta
.PROGRAM paleta()
        SPEED 15 ALWAYS
        MOVE pr[0]
        FOR j = 0 TO 6
            FOR i = 0 TO 5
                IF i < 1 THEN
                    SPEED 100
                END
                MOVE SHIFT(pr[i] BY 0, 56.67*j, 0)
            END
            FOR i = 0 TO 6
                IF i > 1 THEN
                    SPEED 5
                END
                MOVE SHIFT(od[i] BY 0, -j*18, 0)
            END
        END
.END
```

7 Kreiranje SCARA (ADEPT) robota s programom WSPACE

1. Skreirajte SCARA konfiguracijo robota v naravni velikosti po navodilu. Željena pozicija in orientacija koordinatnih sistemov je prikazana na sliki 2.
2. Korigirajte definirani model robota tako, da bodo koordinatni sistemi postavljeni kot na sliki. Translacijsko morate premakniti koordinatne sisteme baze prvega in drugega sklepa.
3. Preizkusite premikanje robota v PENDANT menu z CpMove, Park in TpMove načinih

7.1 Potek vaje

Navodilo za postavitev željene konfiguracije robota smo dobili podano, za primer podajam načrtovanje baze robota:

Vnos v rač.	**Akcija komentar**
I	Input, kurzor premaknemo
-150,0,150	absolutno na to mesto
B	nariši box z imenon Baza in
BAZA	z dimenzijami
+520, 520, 530	(x,y,z)
G	Kreiraj cilinder z imenom
MOTOR	in dimenzijami
175, 8, 280	(radij,št. črt, višina)
M	Move (motor) - z miško ga
	premaknem na ustrezno mesto
Q	Premikamo relativno za
0, 0, 165	(x,y,z)
F8	Zoom out
F4	Pogled od zgoraj

Podobno izračunamo ostale segmente konfiguracije.Še nekaj ukazov ki smo jih uporabljali pri konstrukciji:

F3 pogled s strani

F2 pogled s druge strani

F5 3D pogled

J (attach) združitev dveh elementov

O označimo element na katerega se bodo nanašali ukazi

C (copy),kopiramo element v drugo ime

T (turn) obračamo element

W barvamo element

8 Simulacija gibanja SCARA (Adept) robota s programom WSPACE

8.1 Naloga

1. Robotu dodajte še delovno mizo

2. Robotu definirajte nove min in max vrednosti v sklepih. Uporabite Limits možnost v Model, Define Robot menu

3. Preizkusite delovanje PickAndPlace funkcije, ali uporabite GraspObj, ReleaseObj vunkciji v Action menuju.

4. Programirajte simulacijo dako, da bo robo predmete iz ene palete preložil v drugo in nazaj. Uporabljajte TpMove, GrespObj, ReleaseObj in kreirajte *.trk file ter kot rezultat tudi pic file z željeno simulacijo.

Slika 6: Robot Scara in elementi

8.2 Potek vaje

Cilj naloge je bila simulacija premikanja kock iz ene palete v drugo. Pri tem smo uporabljali:

TearnTp definiramo točko

GraspObj prime predmet

ReleaseObj predmet spusti

Najpomembnejši ukaz pri tej nalogi je PickAndPlace iz ObjBased za prenašanje elementov iz ene palete na drugo

Kreiranje simulacije:

- gremo v SIMULATE meni kjer, WRITE PICS postavimo na 'OFF' to pomeni da se ne vpisuje direktno v *.pic file kjer je končna oblika simulacije ampak v *.trk file
- izberemo željeni model robota
- v ACTION meniju premikamo robota in objekte
- z EDITTRACK editiramo posamezne akcije
- zopet gremo v SIMULATE meni kjer izberemo 'ON' v WRITEPICS nato pa še v tem meniju določimo FILEMAGAIN kjer vnesemo ime *.pic: FILEMAGAIN povzroči da se akcije naučene v startrack ponovijo
- za večkratno ponovitev akcije izberemo LOOP

Ko smo imeli robota na zaslonu smo šli v MODEL meni (DefineRobot, CreateRobot) in tu določili povezave ,sklepe in število prostostnih stopenj. V Exact kinematics smo vpisali prave vrednosti pozicij koordinatnih sistemov.

Preizkus premikanja robota a.)PENDANT meni nam služi kot ročna naprava saj robota lahko premikamo po posameznih sklepih ali pa v globalnih koordinatah:

LEARN Tp robot si zapomni koordinatne točke na zaslonu

DELETE Tp brisanje definiranih točk

FOLLOW Cp robot sledi miški na zaslonu(zadnja dva ukaza delujeta le ,če je inverzna kinematika še vedno prisotna v programu)

b.)ACTION nam doda še dodatne načine premikanja:

Cp MOVE z miško definiramo dodatno točko v prostoru in robot se premakne v to točko

PARK robot se premakne v izhodiščni položaj (HOME procedura)

Ker program ni direktno viden v porgramu *wspace* prilagam datoteko učenja , ki je tipa ASCII in iz katere je razvidno, kako so bile postavljene točke kocke na mizi.

```
teachfile
TRALALA.MOD
MOTOR
```

```
T1
joint,-27.8417,-54.6998,-279.3321,-82.5415,0,0,0,0,0
T2
joint,-25.3105,-69.5121,-279.3321,-94.8227,0,0,0,0,0
NONAME
joint,-89.1165,83.364,-279.3321,-5.7525,0,0,0,0,0
T3
joint,-89.1165,83.364,-279.3321,-5.7525,0,0,0,0,0
T4
joint,-96.1355,95.3434,-279.3321,-0.7921,0,0,0,0,0
T5
joint,-86.1605,85.6318,-156.205,-0.5286,0,0,0,0,0
T11
joint,-40.3172,83.5375,-279.3321,43.2203,0,0,0,0,0
T12
joint,-43.1064,98.8555,-180.0715,55.749,0,0,0,0,0
T13
joint,-41.6861,109.7341,-180.0715,68.048,0,0,0,0,0
T14
joint,-41.0486,125.1194,-180.0715,84.0709,0,0,0,0,0
```

9 Študij Denavit - Hartenbergove notacije in meritev z učnim modelom

9.1 Naloga

1. Preučite Denavit-hartenbergovo (DH) notacijo za zapis kinematičnega modela enega sklepa
2. Zapišite kinematični model za izračun pozicije in orientacije koordinatnega sistema na vrhu gibljivega segmenta učnega modela v skladu z DH notacijo
3. Učni model v izbrani pozi premerite z merilno napravo OPTOTRAK.
   - Kalibrirajte OPTOTRAK s kalibracijsko ploščo , ki mora biti povezana s sistemsko enoto. Plošča naj stoji na mestu, ki je označeno na tleh laboratorija.(direktorij D:
VAJEOR; batch VAJE.BAT; program collect menu Windows / System / Calibrate/ Perform calibration)
   - Odstranite kalibracijsko ploščo in na njeno mesto postavite učni model v pravilni orientaciji.(x os sovpada z orientacijo $\theta = 0$.Na sistemsko enoto sedaj priključite merilne markerje.
   - Merite (menu Windows / optotrak; **F6** naloži ekran Vaje;**F3** prižge/ugasne IR diode).
   - Zapišite si izmerjene vrednosti pozicije in enotinih vektorjev orientacije koordinatnega sistema na koncu segmenta. (Korekcija zaradi pritrditev merilnega okvirja z IR diodami je že upoštevana v programu za prikaz na ekran.)
4. Učni modelmodel segmenta naj ostane v izbrani legi. Z ročnimi merilnimi pripomočki (kovinski meter, kljunasto merilo, kotomer,libela, merilec naklona, merilne palice na modelu) izmerite parametre, ki po DH notaciji zadostujejo za opis kinematičnega modela.
5. Vstavite izmerjene vrednosti v kinematični model in izračunajte numerične vrednosti izbrane pozicije in orientacije koordinatnega sistema na koncu segmenta . Rezultat naj bo matrika 4x4 z ustreznimi numeričnimi vrednostmi.
6. Primerjajte vašo ročno izmerjeno in OPTOTRAK izmerjeno transform acijsko matriko. Ugotovite odstopanja in jih ustrezno komentirajte.

9.2 Potek vaje

Pravilo za tvorjenje Denavit-Hartenbergove notacije izgleda tako:

$$Rot(z, \theta) trans(z, d) trans(x, a) rot(x, \alpha)$$

Slika 7: Denavit-Harenbergova notacija

Po ročni določitvi parametrov smo dobili sledeče rezultate:

$$\theta = 40\deg$$
$$d = 58cm... - 1.75 zaradivisinepodstavka$$
$$a = 18.4cm$$
$$\alpha = 109$$

Pri določitvi HDH smo uporabili program MATHEMATICA ki smo ga pognali z osnovnega direktorija z imenom math .V programu je z"e vsebovan podprogram ki računa HDH notacijo sam brez poprejšnjih množenj matrik.

```
Mathematica (MS-DOS 386/7) 1.2 (June 7, 1990)
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:= Needs["Geometry‘Robotics‘"]
In[2]:= A1=HDHmatrix[theta,d,a,alfa]
Out[2]= HDHmatrix[theta, d, a, alfa]
In[3]:= theta=.33
Out[3]= 0.33
In[4]:= d=566.5
Out[4]= 566.5
In[5]:= a=184
Out[5]= 184
In[6]:= alfa=1.9
Out[6]= 1.9
In[7]:= A1
Out[7]= HDHmatrix[0.33, 566.5, 184, 1.9]
In[8]:= A1=HDHMatrix[theta,d,a,alfa]
```

$$Out_{27} = \begin{bmatrix} 0.764842 & 0.208269 & 0.609623 & 140.731 \\ 0.644218 & -0.247265 & -0.72377 & 118.536 \\ 0. & 0.9463 & -0.32329 & 566.5 \\ 0. & 0. & 0. & 1.0 \end{bmatrix}$$

Matrika ki jo je določil **optotrak** ima približno enake vrednosti kot izračunana matrika

$$\begin{bmatrix} 0.75 & 0.24 & 0.61 & 136.0 \\ 0.66 & -0.21 & -0.72 & 117.5 \\ -0.04 & 0.95 & -0.32 & 546.0 \\ 0 & 0 & 01 & \end{bmatrix}$$

Razlika med izmerjenimi in izračunanimi vrednostni nastopi ker ni bilo mogoče dovolj precizno izmeriti z priročnimi sredstvi kote in razdalje.

10 Izračun matrik direktne kinematike po D-H za robot Scara

- Po D-H anlitično izračunajte direktno kinematiko robota SCARA , konfiguracija RRTR.Pri tem izhajajte iz postavitve koordinatnih sistemov kot jih prikazuje skica. Iz zapisa morajo biti razvidne matrike A in matrika T_4
- Izračunajte matriko T_4 v trenutku ko so kinematični parametri: $a_1 = 425mm, a_2 = 375mm, d_4 = 200mm$ in so sklepi v položajih

$$q = (-0.711rad, 0.1857rad, 473mm, 0.013rad)$$

Slika 8: Kinematika robota Scara

```
Mathematica (MS-DOS 386/7) 1.2 (June 7, 1990)
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin, D. Withoff and T. Sherlock
Copyright 1988,1989 Wolfram Research Inc.

In[1]:= Needs["Geometry`Robotics`"]

In[2]:= A1=HDHMatrix[theta1,0,a1,0]
```

$$Out_2 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & a_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & 0 & a_1\sin\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[3]:= A2=HDHMatrix[theta2, 0, a2, 0]
```

$$Out_3 = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & -\cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[4]:= A3=HDHMatrix[Pi/2, d3, 0, 0]
```

$$Out[4] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[5]:= A_4=HDHMatrix[theta4, d4, 0, 0]
```

$$Out[5] = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & d4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[6]:= T40=A_1.A_2.A3.A_4
```

$$\begin{bmatrix} \cos\theta_4\,(\cos\theta_2\sin\theta_1 + \cos\theta_1\sin\theta_2) - (\cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2)\sin\theta_4 & -(\cos\theta_4\,(\cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2)) - (\cos\theta_2\sin\theta_1 + \cos\theta_1\sin\theta_2)\sin\theta_4 & 0 & a_1\cos\theta_1 + a_2\cos\theta_1\cos\theta_2 - a_2\sin\theta_1\sin\theta_2 \\ \cos\theta_4\,(-(\cos\theta_1\cos\theta_2) + \sin\theta_1\sin\theta_2) - (\cos\theta_2\sin\theta_1 + \cos\theta_1\sin\theta_2)\sin\theta_4 & -(\cos\theta_4\,(\cos\theta_2\sin\theta_1 + \cos\theta_1\sin\theta_2)) - (-(\cos\theta_1\cos\theta_2) + \sin\theta_1\sin\theta_2)\sin\theta_4 & 0 & a_1\sin\theta_1 + a_2\cos\theta_2\sin\theta_1 + a_2\cos\theta_1\sin\theta_2 \\ 0 & 0 & -1 & -d_3 - d4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[7]:= T4=SimplifyTrig[T40]
```

$$\begin{bmatrix} s(\theta_1+\theta_2-\theta_4) & -s(\theta_1+\theta_2-\theta_4) & 0 & a_1c\theta_1 + a_2c(\theta_1+\theta_2) \\ -c(\theta_1+\theta_2-\theta_4) & -s(\theta_1+\theta_2-\theta_4) & 0 & a_1s\theta_1 + a_2s(\theta_1+\theta_2) \\ 0 & 0 & -1 & -d_3 - d4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[8]:= a1=425
Out[8]= 425
In[9]:= a2=375
Out[9]= 375
In[10]:= d4=200
Out[10]= 200
In[11]:= theta1=0
Out[11]= 0
In[12]:= theta2=0
Out[12]= 0
In[13]:= d3=437
Out[13]= 437
In[14]:= theta4=0.0
Out[14]= 0.
In[15]:= T4
```

$$Out[15] = \begin{bmatrix} 0. & -1. & 0 & 800 \\ -1. & 0. & 0 & 0 \\ 0 & 0 & -1 & -637 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
In[16]:= theta1=-0.711
Out[16]= -0.711
In[17]:= theta2=1.857
Out[17]= 1.857
In[18]:= theta4=0.013
Out[18]= 0.013
In[19]:= T4
```

$$T4 = \begin{bmatrix} 0.905688 & -0.423945 & 0 & 476.577 \\ -0.423945 & -0.905688 & 0 & 64.3195 \\ 0 & 0 & -1 & -637 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$