

Robòtica: Raiden Project

Bartomeu Miró Mateu *

Lluís Cortès Rullan †

3 de maig de 2011

Primera pràctica de Robòtica de l'apartat de robòtica industrial. Control d'un braç robot Mitsubishi RV-6S programat en MELFA Basic IV destinant a la paletització de peces colorides circulars.

*bmiro a members punt fsf punt org

†lluisbinet a gmail punt com

Índex

1 Interpretació de l'enunciat i modelat de l'escenari

En aquest apartat s'expliquen les extensions i suposicions de l'enunciat original, així el modelat de l'entorn del robot.

Tal i com es demana el programa està parametritzat segons la posició de la càmera, el punt P que aquesta detecta i l'angle α . A més existeixen altres paràmetres com els punts on es troben les peces originalment i on es deixen al final.

Per altra banda s'ha introduït la possibilitat de fixar un nombre diferent de peces a cada pila, per tant es poden tenir piles amb diferent número de peces. També és té amb compte la possibilitat de tenir més d'una peça, o cap, d'algun tipus.

Tots aquests paràmetres es tornen a veure detallats en l'apartat d'explicació del codi on s'expliquen les *Macros pròpies*??.

En la figura ?? següent es poden veure tots els paràmetres i punts (marcats amb una estrella) que caracteritzen l'escenari.

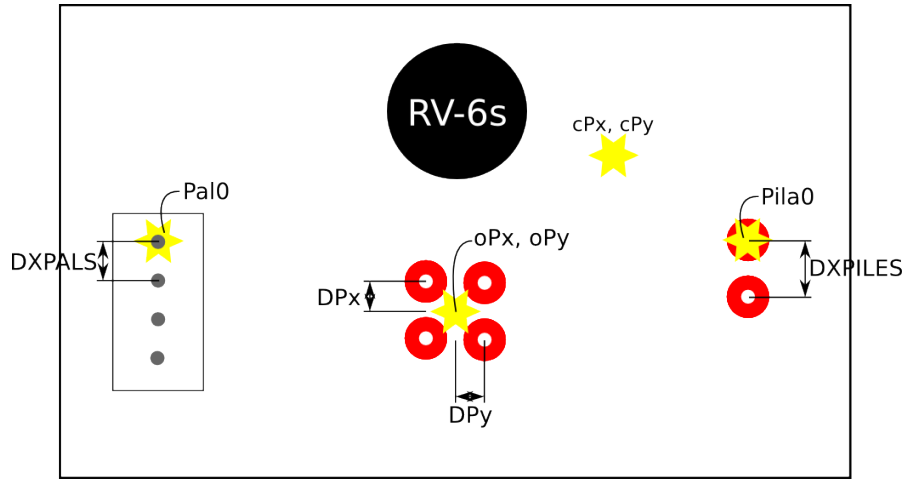


Figura 1: Escenari del robot

La figura es veu complementada amb el llistat de punts emprats a la pràctica.

DEF POS Paralisi	=	(337.16, -14.95, 270.30, -179.98, -0.36, -177.39) (7, 0)
DEF POS Pa0	=	(315.50, -550.72, 280.00, -173.00, -67.00, -90.00) (7, 0)
DEF POS Pila0	=	(337.18, 486.51, 280.00, -179.98, -0.36, -90.00) (7, 0)
DEF POS Pale0	=	(0.00, 0.00, 280.00, -179.98, -0.36, -90.00) (7, 0)
DEF POS PalePT	=	(0.00, 0.00, 280.00, 100.41, 85.49, 98.72) (6, 0)
DEF POS PaleOut0	=	(337.00, -450.00, 280.00, -172.92, -66.76, -90.00) (7, 0)

Com es detalla en l'apartat de *calcul de punts* ?? s'ha intentat minimitzar el nombre de punts capturats per tal de calcular els demes en realció a aquests.

A continuació s'explica l'utilitat de cada punt o posició.

Paralisi Guarda la posició del robot en repòs.

Pal0 Posició del primer pal (amb la pinça tombada).

Pila0 Posició de la primera pila (amb pinça perpendicular)

Pale0 Orientació del braç robot per posar peces del palé (amb la pinça perpendicular).

PaleOut0 Orientació del braç robot per agafar les peces del palé (amb la pinça tombada)

L'enunciat deixa oberta la possibilitat de que fer amb les peces de tipus 4, en aquest punt s'ha optat per posar-les al pal 4 per aprofitar el codi ja escrit i així seguir la coherència i estructura dels tipus de peça anteriors. El fet de no optar per apilar-les en qualsevol punt de l'entorn es perquè en la paletització ja s'ha demostrat coneixement de com apilar peces i tractar el tipus 4 de manera diferent als anteriors minvava elegància al codi i l'execució.

A efectes pràctics a la imatge següent es veu el posicionat de les peces de una forma humanament comprensible emprant les marques fetes per els alumnes sobre l'entorn i existents a data de 3 de maig de 2011.

TODO FOTO peces sobre l'entorn.

A la imatge es pot apreciar com la primera pila esta al centre fet amb rotulador negre i la segona seguint el cercle fet amb boli *Bic* blau.

2 Moviment del robot

En aquest punt descrivim el moviment del braç robot i el perquè de l'ordre o orientacions del mateix.

Totes les aproximacions a les peces es fan desde adalt. El braç robot es posiciona a XY sobre la peça en qüestió i després efectua un descens. Un cop agafada efectua el moviment invers, ascens a Z i després el desplaçament corresponent al pla XY on es consideren segurs els moviments. Aquest pla de seguretat en la practica ve donat per la variable ZS fixada per defecte a 280.0.

En primer lloc el robot agafa les peces del les piles inicials. Aquestes són col·locades a la zona de paletització. Ambudes accions és duen a terme amb la pinça perpendicular al pla XY, ja que es la posició més segura i còmode de programar per l'agafada de peces.

Un cop acabat el muntatge del palé es desmunta amb la pinça tombada, s'agafen les peces començant per les que es troben més a l'esquerra del braç robot, per col·locarles als pals.

El fet de tombar la pinça és per incrementar l'abast del braç robot. Amb la pinça perpendicular al pla XY, com s'havia efectuat el muntatge del palé, no és possible arribar als pals 3 i 4. Com que s'ha decidit tombar la pinça cap a l'esquerra¹, desde el punt de vista del robot, s'han de recollir les peces d'esquerra a dreta per tal de evitar que el braç col·lisió amb algun dels munts del palé encara existents.

¹Com que els pals estan a la dreta del robot convé tombar la pinça a la esquerra per simplificar els moviments i evitar haver de fer un gir del braç

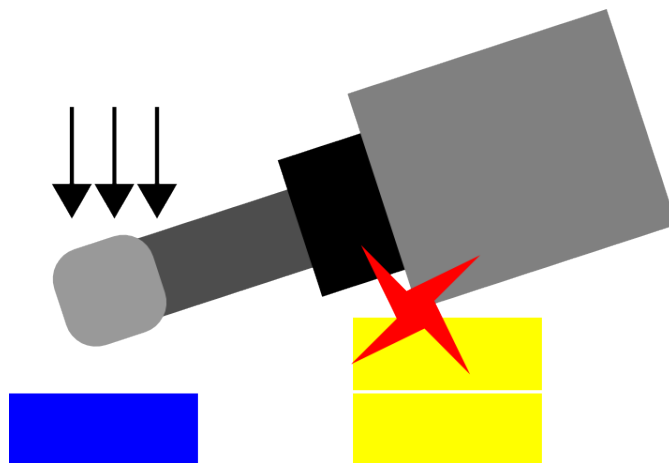


Figura 2: Colisió amb el munt de peces grogues intentant agafar l'última peça blava

Així doncs a la figura?? veim quin seria l'ordre de recollida depenent dels diferents angles per tal d'evitar les possibles col·lisions.

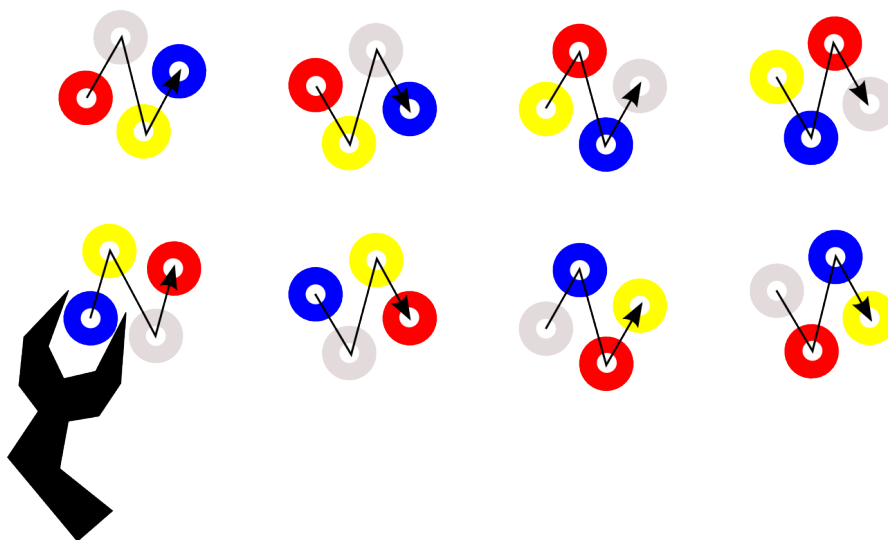


Figura 3: Ordre de recollida de les peces segons l'angle α

Així doncs l'ordre de despaletització depèn de l'angle i no del tipus de peça. Com és pot veure a la figura existeixen 8 possibles casos que es veuen reflectits en l'explicació del codi font corresponent, secció ??.

3 Càlculs

L'apartat de càlculs inclou, per una banda, els requerits per l'encunyat a l'haver de transformar el punt P del sistema de coordenades de la càmera al del braç robot. En aquest punt es mostra el procés de obtenció de les equacions per fer tal transformació i en l'apartat de codi es veu la seva implementació en el programa.

Per altra banda es mostra el càlcul dels punts emprats de la situació dels objectes en l'entorn a partir dels seus homòlegs. Com s'ha mencionat la posició dels pals es relativa a un punt, concretament tots els pals on es deixen les peces venen en funció de la posició del primer. De la mateixa manera la posició de cada pila on inicialment es recullen les peces ve en funció de la posició de la primera pila, el nombre de piles i la distància entre elles. Finalment també es descriu el càlcul de punts en Z on s'agafen les peces en funció del pla de terra, l'altura de les peces i la quantitat de les mateixes.

3.1 Transformació de sistemes de coordenades

Amb l'enunciat tenim la posició de la càmera {C} respecte el robot {R} i la posició del punt P, que es el centre de sistema de coordenades del palé {P}, respecte la càmera. El que volem es posar el punt P en el sistemes de coordenades del robot.

Així doncs definim:

${}^R T_C$ La matriu de transformació de un punt del robot a la càmera

${}^C T_P$ La matriu de transformació de un punt de la càmera al palé

${}^P P$ Punt en el sistema de coordenades del palé

${}^R P$ Punt en el sistema de coordenades del robot

Combinant aquestes matrius obtenim que:

$${}^R T_C \times {}^C T_P \times {}^P P = {}^R P$$

Per tant hem de cercar ${}^R T_P$:

$${}^R T_P = {}^R T_C \times {}^C T_P$$

$${}^R T_P = \left(\begin{array}{ccc|c} 0 & 1 & 0 & cPx \\ 1 & 0 & 0 & cPy \\ 0 & 0 & -1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right) \times \left(\begin{array}{ccc|c} \cos\phi & -\sin\phi & 0 & oPx \\ \sin\phi & \cos\phi & 0 & oPy \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right)$$

$${}^R T_P = \left(\begin{array}{ccc|c} \sin\phi & \cos\phi & 0 & oPy + cPx \\ \cos\phi & -\sin\phi & 0 & oPx + cPy \\ 0 & 0 & -1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right)$$

Així doncs per passar d'un punt del sistema de coordenades ${}^P P$ a ${}^R P$ ho feim així:

$${}^R T \times {}^P P = {}^R P$$

En el cas generic agafam ${}^P P = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$

$$\left(\begin{array}{ccc|c} \sin\phi & \cos\phi & 0 & oPy + cPx \\ \cos\phi & -\sin\phi & 0 & oPx + cPy \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right) \times \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \times \sin\phi + y \times \cos\phi + cPy + cPx \\ x \times \cos\phi - y \times \sin\phi + oPx + cPy \\ -z \\ 1 \end{pmatrix}$$

De on extreim les següents equacions que seran implementades dins del nostre programa??.

$${}^R P_X = x \times \sin\phi + y \times \cos\phi + cPy + cPx$$

$${}^R P_Y = x \times \cos\phi - y \times \sin\phi + oPx + cPy$$

$${}^R P_Z = -z$$

3.2 Càlcul de piles, pals (X,Y) i altura de peces (Z)

A banda de la transformació de punts entre sistemes de coordenades hi ha altres punts a calcular. El primer són les piles d'on s'agafen les peces. Dins del llistat de punts tenim el punt de la primera pila, les següents es troben situades a un desplaçament en X. Així doncs en el nostre escenari senzillament s'ha de incrementar en X el valor de la distancia entre piles. Succeeix exactament el mateix amb els pals on es deixen les peces.

Els detalls de implementació i valors de les constants de separació es poden veure en l'apartat específic ?? on s'explica el codi font. Cal remarcar que al codi també figura un desplaçament en Y, per si es volguessin posar les piles o pals fent una diagonal, però la *macro* està inicialitzada a 0.

4 Estructura del programa

El programa es divideix en cinc seccions.

Capçalera de l'enunciat Interfície de macros requerida per l'enunciat on hi ha paràmetres com l'angle α .

Macros pròpies Reassignació de valors per les *features* extra que s'han implementat com ara emprar piles amb nombre de peces diferents, si es vol fer una execució amb l'enunciat original sols fa falta assignar els valors de les variables de la secció anterior.

Declaracions pròpies Aquí es declaren les variables globals pròpies del programa, en principi sols les ha de tocar el programador.

Rutines Codi de les rutines on es desenvolupa tota la pràctica.

Main Simple crida ordenada a les rutines per desencadenar l'execució.

4.1 Macros, paràmetres

Les *macros* de la *capçalera de l'enunciat* ja estan explicades al propi enunciat, aquí s'expiquen les altres *macros* introduïdes per les *features* pròpies, corresponents a l'apartat de *Macros pròpies*.

Cal mencionar aquests valors no són realment macros sinó variables, una macro es resol en temps de preprocessador no en temps de compilació o execució, aquest fet prota alguns problemes mencionats a l'apartat de incidents TODOLABEL.

PINCA Valor de la pinça emprada, en el nostre cas el robot només en té una i es la 1.

DPALE Retard de 4 segons requerit a l'enunciat un cop s'ha muntat el palé.

D#PINCA% Retards aplicats després de O#brir o T#ancar la pinça en funció de si es I%ncial o F%inal, abans o després d'agafar la peça.

VLENT V_elocitat a la qual es duen a terme les aproximacions delicades, com l'agafada de peces.

VNORMAL V_elocitat a la que es desplaça normalment el braç en els trajectes segurs.

ZS Pla Z on es consideren segurs els moviments del braç robot i on es duen a terme els desplaçaments llargs.

ZTPT Pla de T_erra ($Z = 0$) amb la P_inça T_ombada (angle inferior a 45 amb el pla XY)

ZTPR Pla de terra amb la pinça perpendicular al pla XY

TP T_ipus de P_eça diferents que tenim, es igual el numero de munts del palé i de pals de destí.

PILES Número de piles d'on es fa la recollida inicial de peces.

PALS Número de pals de destí on es deixen les peces.

DT Retard per l'obertura i tancament de la pinça.

HDISC Altura de les peces (discs)

D#PILES Distància en X i Y entre les piles

D#PALS Distància en X i Y entre els pals

DZPAL Distància Z a baixar per introduir la peça dins el pal.

RELPALE(T,E) Distància relativa de la peça tipus T al punt P, en funció de l'eix E (1 per X, 2 per Y)

4.2 Estructura del codi

En el codi s'ha fet una funció per cada acció i alguna d'auxiliar transversal. La intenció es dedicar tota la feina a les rutines i que la lectura del programa es pugui llegir de manera natural en els nivells més alts entrant en detalls a mesura que es va aprofundint en les funcions.

Munta palé		Desmunta palé	
Agafa pila	Posa palé	Agafa Palé	Posa pal
Obre/tanca pinça			

Cal fer especial menció a dos vectors de control de l'estat del programa, aquests son `npcspila` i `alloc`.

`npcspila(p)` conté el nombre de peces que resten a la pila inicial de on s'agafen les peces.

`alloc(m)` indica quantes peces ja estan a lloc, situades al munt `p` del palé, recordem que cada munt correspon a un tipus de peça.

Ambdues variables serveixen per controlar l'altura en `Z` d'on agafar les peces, i per saber quan ja no en queden més.

Amb aquest principi ens queda un cos principal tan lleguer com aquest.

```
960 gosub *INIT
970 gosub *CALCPTS
980 gosub *MNTPALE
990 dly DPALE
1000 gosub *DESPAILE
1010 gosub *ACABA
1020 end
```

4.2.1 Auxiliars

Canvi sistemes de coordenades En la funcio `CAM2ROB` funció es troba implementat el canvi de sistemes de coordenades del la camera al robot. Com a interfície llegeix els punts `ppx` i `ppy` que son els que es desitgen tranformar, així com les macros del programa `oPx`, `oPy`, `cPx`, `cPy` i `l'angle`. Deixa els valors de `ppx` i `ppy` en funció del sistema de coordenades del braç robot en les variables `prx` i `pry`.

```
2010 *CAM2ROB
2020     angleR =(angle*M_PI) / 180.0
2030     prx = ppx*sin(angleR) + ppy*cos(angleR) + oPy + cPx
2040     pry = ppx*cos(angleR) - ppy*sin(angleR) + oPx + cPy
2050     return
```

Obertura-Tancament de pinça Per tal de facilitar la llegibilitat del codi i no afegir constantment els retards per la obertura i tancament de pinça s'han definit les funcions OPINCA i TPINCA que encapsulen aquest comportament.

```
2070 *OPINCA
2080     dly DOPINCAI
2090     hopen PINCA
2100     dly DOPINCAF
2110     return
```

```
2120 *TPINCA
2130     dly DTPINCAI
2140     hclose PINCA
2150     dly DTPINCAF
2160     return
```

4.2.2 Inicialitzacions

En aquesta rutina es duu a terme l'operació de *homing* i demés inicialitzacions físiques del braç robot com l'obertura de la pinça o de les variables que controlen l'entorn del robot, com el contador de peces que s'han posat a cada palé.

```
1040 *INIT
1050     servo ON
1060     ovrd VNORMAL
1070     mov Paralisi
1080     gosub *OPINCA
1090     for i = 1 to TP
1100         alloc(i) = 0
1110     next
1120     return
```

4.2.3 Càlcul de punts

Com ja s'ha comentat en la pràctica s'ha fet el màxim esforç per posar tots els punts en funció d'uns pocs, per això en la funció de càlcul de punts s'omplen els vectors que contenen els punts homòlegs calculats a partir de les *macros* que descriuen l'entorn físic. En aquest punt també es calcula l'ordre de recollida de les peces en funció de l'angle2. Per limitacions del llenguatge comentades a l'apartat de incidents TODO LABEL el codi no has sortir gaire elegant.

Un cop acabada la funció càlcul de punts tots els vectors de punts estan apunt per ser recorreguts per part del braç robot en els eixos (X,Y), el Z és calculat en temps d'execució ja que depèn de quina peça s'estigui tractant, ja que son apliades a l'eix Z. Aquests vectors emprats de interfície són:

Pila Cada component indica la posició (X,Y) de cada una de les piles d'on s'agafen inicialment les peces, en l'execució per defecte seran dues.

Pale Cada component correspon a la posició on anirà col·locat cada tipus de peça al palé.

PalI Posició inicial on es col·loca el braç per iniciar el descens sobre el pal.

PalF Punt final de descens del braç, on es considera que la peça ja hi ha entrat i pot ser amollada.

```
1130 *CALCPTS
1140   for pil = 1 to PILES
1150     Pila(pil) = Pila0
1160     Pila(pil).x = Pila(1).x + DXPILES * (pil - 1)
1170     Pila(pil).y = Pila(1).y + DYPILES * (pil - 1) 'es 0
1180   next

1200   for tipus = 1 to TP
1210     ppx = RELPALE(tipus, 1) 'parametres de entrada de CAM2ROB
1220     ppy = RELPALE(tipus, 2)
1230     gosub *CAM2ROB          'prx i pry son parametres de sortida
1240     Pale(tipus) = Pale0     'Escriu altres components i orientacions
1250     Pale(tipus).x = prx
1260     Pale(tipus).y = pry
1270     PaleOut(tipus) = PaleOut0
1280     PaleOut(tipus).x = prx + 5.0
1290     PaleOut(tipus).y = pry -20.0
1300   next

1320   for pal = 1 to PALS
1330     PalI(pal) = Pal0
1340     PalI(pal).x = PalI(pal).x + DXPALS * (pal - 1)
1350     PalI(pal).y = PalI(pal).y + DYPALS * (pal - 1)
1360     PalF(pal) = PalI(pal)
1370     PalF(pal).z = PalF(pal).z - DZPAL
1380   next
1390
1400   angle = angle mod 360
1410   if ((0.0 < angle) and (angle < 45.0)) then
1420     ordre(1) = 3
1430     ordre(2) = 2
1440     ordre(3) = 4
1450     ordre(4) = 1
1460     return
1470   endif
1480   if ((45.0 < angle) and (angle < 90.0)) then
1490     ordre(1) = 3
```

```

1500         ordre(2) = 4
1510         ordre(3) = 2
1520         ordre(4) = 1
1530         return
1540     endif
1550     if ((90.0 < angle) and (angle < 135.0)) then
1560         ordre(1) = 4
1570         ordre(2) = 3
1580         ordre(3) = 1
1590         ordre(4) = 2
1600         return
1610     endif
1620     if ((135.0 <= angle) and (angle < 180.0)) then
1630         ordre(1) = 4
1640         ordre(2) = 1
1650         ordre(3) = 3
1660         ordre(4) = 2
1670         return
1680     endif
1690     if ((180.0 <= angle) and (angle < 225.0)) then
1700         ordre(1) = 1
1710         ordre(2) = 4
1720         ordre(3) = 2
1730         ordre(4) = 3
1740         return
1750     endif
1760     if ((225.0 <= angle) and (angle < 270.0)) then
1770         ordre(1) = 1
1780         ordre(2) = 2
1790         ordre(3) = 4
1800         ordre(4) = 3
1810         return
1820     endif
1830     if ((270.0 <= angle) and (angle < 315.0)) then
1840         ordre(1) = 2
1850         ordre(2) = 1
1860         ordre(3) = 4
1870         ordre(4) = 3
1880         return
1890     endif
1900     ' 315.0 > angle < 360.0
1910     ordre(1) = 2
1920     ordre(2) = 3
1930     ordre(3) = 1
1940     ordre(4) = 4
1950     return

```

4.2.4 Munta palé

Tal i com indica el títol en aquesta funció es munta el palé, aquesta tasca consisteix en iterar per cada una de les peces existents en cada una de les piles per agafar-les i depositar-les al lloc corresponent del palé.

```
2170 *MNTPALE
2180     peca = 1
2190     for pil = 1 to PILES
2200         for pecapila = 1 to npcspila(pil)
2210             gosub *AGAFPILA
2220             gosub *POSAPALE
2230             peca = peca + 1
2240         next
2250     next
2260     return
```

Agafa de la pila

```
2370 *AGAFPILA
2380     mov Pila(pil)
2390     Prvsnl = Pila(pil)
2400     Prvsnl.z = Prvsnl.z - ZTPR + (npcspila!(pil) - pecapila) * HDISC
2410     ovrd VLENT
2420     mvs Prvsnl
2430     gosub *TPINCA
2440     mvs Pila(pil)
2450     ovrd VNORMAL
2460     return
```

Posa al Palé

```
2470 *POSAPALE
2480     mov Pale(tipusP(peca))
2490     Prvsnl = Pale(tipusP(peca))
2500     Prvsnl.z = Prvsnl.z - ZTPR + alloc(tipusP(peca)) * HDISC
2510     ovrd VLENT
2520     mvs Prvsnl
2530     gosub *OPINCA
2540     mvs Pale(tipusP(peca))
2550     ovrd VNORMAL
2560     alloc(tipusP(peca)) = alloc(tipusP(peca)) + 1
2570     return
```

4.2.5 Desmunta palé

En aquest punt del programa el braç va munt a munt del palé agafant les peces i deixant-les als pals, no passa al següent munt mentre a l'actual quedin peces. L'ordre d'agafada dels munts ha estat calculat en *calcul de punts* com ja s'ha esmentat??.

Agafa del palé

```
2580 *AGAFPALE
2590     mov PaleOut(munt)
2600     Prvsnl = PaleOut(munt)
2610     Prvsnl.z = Prvsnl.z - ZTPT + (alloc(munt) - 1) * HDISC
2620     ovrd VLENT
2630     mvs Prvsnl
2640     gosub *TPINCA
2650     mvs PaleOut(munt)
2660     ovrd VNORMAL
2670     alloc(munt) = alloc(munt) - 1
2680     return
```

Posa pal

```
2690 *POSADEST
2700     mov PalI(munt)
2710     ovrd VLENT
2720     mvs PalF(munt)
2730     gosub *OPINCA
2740     mvs PalI(munt)
2750     ovrd VNORMAL
2760     return
```

4.2.6 Acaba

De manera similar a l'inicialitza, acaba s'encarrega del posicionament físic del robot en una posició de repòs i la desconnexió dels servomotors.

```
1960 *ACABA
1970     mov Paralisi
1980     hopen PINCA
1990     servo OFF
2000     return
```

5 Incidents i problemàtica al desenvolupament

tombar pinça etc.

Queixes del llenguatge i entorn case i float
falta de macros, sobretot per declaració de vectors
brutor dels gosub i variables globals com a parametre
offset amb la pinça tombada
les peces s'aferren entre elles per pressio i restes de cola o a la cinta de la pinça quan aquesta esta desgastada.

6 Joc de proves

totes les peces de un mateix color mira com s'apilen tant la desviacio en xy com l'escas de pressio en z