ARPRO - IRP - Lab n°1

Matthieu LAMIELLE
Muruo WANG

Exercice 1

Code version 1 :

```
edit MLMU1.version1
; Matthieu LAMIELLE et Muruo WANG

;inputs
part.det.P1 = 1004
part.det.P2 = 1005
part.det.R2 = 1008
part.det.R1 = 1007

;output
conveyor.start = 1
conveyor.stop = -1
conveyor.g = -2
conveyor.d = 2

signal 5
tool succion.pad

;Tool check
openi

;Debut
move wait.loc

;On vérifie qu'une pièce est en place
wait sig(part.det.P1) or sig(part.det.P2)

;On regarde où est située la pièce à prendre
if sig(part.det.P1)then
  set P=P1
else
  set P=P2
end

;On va la chercher
appro P, 100
moves P
closei
departs 50

;On la pose sur le tapis
appro place,50
moves place
openi
departs 100

;On le fait partir vers la droite
signal(conveyor.d)
signal(conveyor.start)

;Dès que la pièce est au detector 2 on stop le tapis
wait sig(part.det.R2)
signal(conveyor.stop)

;On attend
timer 1 = 0
wait timer(1) > 1

;On redémarre dans l'autre sens
signal(conveyor.g)
signal(conveyor.start)

```

```
62
63 ;Quand la pièce est au detector 1 on stop le tapis
64 wait sig(part.det.R1)
65 signal(conveyor.stop)
66
67 ;On prend la pièce sur le tapis (à detector 1)
68 appro grasp.loc.stopped, 100
69 moves grasp.loc.stopped
70 closei
71 departs 50
72
73 ;On pose la pièce en P3
74 appro P3,50
75 moves P3
76 openi
77 departs 100
78
79 move wait.loc
80
81 e
82
83 load pickconv.lc
84 load alltools.v2
```

Code version 2 :

```
1  edit MLMU1.version2
2  ; Matthieu LAMIELLE et Muruo WANG
3
4  ;inputs
5  part.det.P1 = 1004
6  part.det.P2 = 1005
7  part.det.R2 = 1008
8  part.det.R1 = 1007
9
10 ;output
11 conveyor.start = 1
12 conveyor.stop = -1
13 conveyor.g = -2
14 conveyor.d = 2
15
16 signal 5
17 tool succion.pad
18
19 ;Tool check
20 openi
21
22 speed 30 always
23 speed 100 mmps always
24
25 ;Debut
26 move wait.loc
27
28 ;On vérifie qu'une pièce est en place
29 wait sig(part.det.P1) or sig(part.det.P2)
30
31 ;On regarde où est située la pièce à prendre
32 if sig(part.det.P1)then
33   set P=P1
34 else
35   set P=P2
36 end
```

```
37
38 ;On va la chercher
39 appro P, 100
40 moves P
41 closei
42 departs 50
43
44 ;On la pose sur le tapis
45 appro place,50
46 moves place
47 openi
48 departs 100
49
50 ;On place le robot en attente
51 appro grasp.loc.stopped, 150
52
53 ;On fait partir le tapis vers la droite
54 signal(conveyor.d)
55 signal(conveyor.start)
56
57 ;Dès que la pièce est au detector 2 on commence le timer
58 wait sig(part.det.R2)
59 timer 1 = 0
60 wait timer(1) > 3
61
62 ;On redémarre dans l'autre sens après un delais de 1s
63 signal(conveyor.stop)
64 signal(conveyor.g)
65 timer 1 = 0
66 wait timer(1) > 1
67 signal(conveyor.start)
68
69 ;On remet le timer à 0 au passaage du detector 2
70 wait sig(part.det.R2)
71 timer 1 = 0
72
73 ;Quand la pièce est au detector 1 on calcule la vitesse
74 wait sig(part.det.R1)
75 vitesse = 700/timer(1)
76 speed (vitesse*sqrt(2)) mmps
77
78 ;On prend la pièce sur le tapis (à detector 1)
79 set grasp.loc.mov = shift(grasp.loc.stopped by 0, 150, 0)
80 openi
81 moves grasp.loc.mov
82 closei
83 departs 50
84
85 ;On pose la pièce en P3
86 appro P3,50
87 moves P3
88 openi
89 departs 100
90
91 move wait.loc
92
93 e
94
95 load pickconv.lc
96 load alltools.v2
```

Questions :

What are the reasons for your program to fail ?
It takes time for the robot to accelerate at its ordered speed. So the cylinder goes too far and the succion pad comes too late.

How to teach the grasp.loc.stopped point to the robot ?
After the execution of the program, when we have to take back the cylinder, we can use the remote control to reach the aimed position (grasp.loc.stopped)

Exercice 2

```
edit ex2
;Matthieu LAMIELLE et Muruo WANG

;Initialisation des outils
tool pen
openi

;Initialisation position du robot
moves wait.loc
signal 5

;inputs
glue.empty = 1009
glue.recharged = 1010

;recharge le glue
reacti glue.empty,recharger,50


appro board:A, 50
moves board:A
closei

moves board:B
break

moves board:C
break

moves board:D
break

moves board:A
break

departs 50

move wait.loc

set board = board:shift(null by 10,10,0)

openi

e

edit recharger

;inputs
glue.recharged = 1010

;On repère le point où on s'est interrompu
set interruption = here
set arrival = dest
openi
departs 50
;On va recharger
appro recharge, 50
moves recharge
wait sig(glue.recharged)
departs 50

```

```
61
62 ;On revient à l'endroit où on s'était arrêté
63 appro interruption, 50
64 moves interruption
65 closei
66
67 ;On finit l'instruction a laquelle on s'était arrêté
68 moves arrival
69
70 e
71
72 load glue.lc
73 load alltools.v2
```

Questions :

1.Should the glue default event be handled synchronously?
We have to detect whether there is glue left or not during all the process even between two points, so we use the synchronous mode.
2.Which are the Val/V+ instructions that can be used?
We can use “reacti” inside our program or run a parallel program to check glue using execute/abort in the main function.
3.Which one will you choose and why?
“reacti” is adapted because we can stop and go back at the point we left after recharging.
4.What is the solution to make sure that the object for which a glue occurred is correctly glued along the whole trajectory?
We have to record both the position and destination of the tool when it runs out of glue, so we can go back to the exact location.

Exercice 3

```
edit ex3
;Matthieu LAMIELLE et Muruo WANG

;Tool
tool gripper

;Initialisation
signal 5

;Définir le frame
set Palette = frame(origin,pt.x,pt.xy,origin)

move wait.loc
openi

i=0
j=0

for i=0 to 2
    for j=0 to 1

        set P = shift(null by 82*i, 59*j, 0)
        set drop.loc = shift(drop.loc by 0, 0, 15)

        appro palette:P, 50
        moves palette:P
        closei
        departs 50

        appro drop.loc, 50
        moves drop.loc
        openi
        depart 50
    end
end
move wait.loc
e
load alltools.v2
load pallet.lc
```

Exercice 4

Code version 1 :

```
edit ex4V1
;Matthieu LAMIELLE et Muruo WANG

speed 30 always
speed 100 mmps always

tool null
stop.condition = 1009

;On bouge manuellement le robot dans sa position de depart
set P = here

;On definit la distance totale à parcourir
distancetot = 200

i = 0

;On definit la longueur du déplacement intermediaire
smalld = 5

timer 1 = 0

do
    set etape = shift(P by 0,0,-i*smalld)
    moves etape
    i = i+1
until(sig(stop.condition) or (i*smalld > distancetot))

vitesse = (i*smalld)/timer(1)
type "La vitesse moyenne est de ", vitesse

e
```

Code version 2 :

```
edit ex4V2
;Matthieu LAMIELLE et Muruo WANG

;Movement gardé
execute 1 parallele, -1

tool null
speed 30 always
speed 100 mmps always

;On bouge manuellement le robot dans sa position de depart
set P = here
distancetot = 200

timer 1 = 0
;Decendre
set fin = shift(P by 0,0,-distancetot)
moves fin
break

vitesse = (distancetot)/timer(1)
type "La vitesse moyenne est de ", vitesse

abort 1

e

edit parallele

;On definit une condition d'arrêt
stop.condition = 1009

if sig(stop.condition) then
    brake
end

e
```

Which solution is the best ?

The parallel program works the best because it doesn't interrupt the calculation during each small displacement.

Why is the average speed always lower than the desired speed ?

For both program, the maximum speed isn't reached instantaneously so the average speed is lower than the maximum speed. In addition for the first program we had a condition to be tested for each displacement so the speed is slower than the improved program.