Report for IRP Labs

Students
Lucia Bergantin
Amr Nassef
Michele Tartari

Attended Labs with Group A

- Exercise 1 Part-1

Code

```
edit ex1

tool succion.pad

sense.p1 = 1004
sense.p2 = 1005

detector1 = 1007
detector2 = 1008

conveyor.start = 1
conveyor.stop = -1
conveyor.go.right = 2
conveyor.go.left = -2

air.pressure = 5
signal air.pressure

wait sig(sense.p1) or sig(sense.p2)
if sig(sense.p1) then
        set obj.pos=p1
else
if sig(sense.p2) then
        set obj.pos=p2
end
end

appro obj.pos,50
moves obj.pos
closei
departs 50
appro place,50
moves place
openi
depart 50
move wait.loc
break
```

```
signal conveyor.start, conveyor.go.right
wait sig(detector2)
signal conveyor.stop
timer 1 = 0
wait timer(1)> 1
signal conveyor.start, conveyor.go.left
wait sig(detector1)
signal conveyor.stop

appro grasp.loc.stopped, 50
moves grasp.loc.stopped
closei
departs 50
appro p3,50
moves p3
openi
departs 50
move wait.loc

e

load alltools.v2
load pickconv.lc
```

Comments

We handled the inputs synchronously for part-1

- Exercise 1 Part-2

Code

```
edit ex1b

tool succion.pad

sense.p1 = 1004
sense.p2 = 1005
detector1 = 1007
detector2 = 1008
conveyor.start = 1
conveyor.stop = -1
conveyor.go.right = 2
conveyor.go.left = -2
air.pressure = 5
conveyor.lenght = 700

speed 30 always
speed 100 mmps always

signal air.pressure
wait sig(sense.p1) or sig(sense.p2)
if sig(sense.p1) then
    set obj.pos=p1
else
if sig(sense.p2) then
    set obj.pos=p2
end
end

appro obj.pos,50
moves obj.pos
closei
departs 50
appro place,50
moves place
openi
depart 50
appro grasp.loc.stopped, 150

signal conveyor.start, conveyor.go.right
wait sig(detector2)
timer 1 = 0
wait timer(1)> 1
signal conveyor.stop
timer 1 = 0
wait timer(1)> 1
signal conveyor.start, conveyor.go.left
wait sig(detector2)
timer 1 = 0
wait sig(detector1)
time.taken = timer(1)

conveyor.speed =conveyor.lenght/time.taken
set obj.move.pose = shift (grasp.loc.stopped by 0,150,0)
speed conveyor.speed*sqrt(2) mmps
moves obj.move.pose
closei
departs 50
appro p3,50
moves p3
openi
departs 50
move wait.loc

e

load alltools.v2
load pickconv.lc
```

Comments

We handled the inputs synchronously. In this case the robot isn't able to catch the part though, due to the handling of velocity. It's essential to let the part pass the second detector before changing the direction of the conveyor because we need to consider the space for the conveyor to accelerate to full speed. However the task still fails due to the lack of compensation of the acceleration of the end effector, that takes a certain amount of time to reach full speed.
To teach the grasp.loc.stopped we have to stop the conveyor when we detect the object. Although we have to consider the diameter of the object when we are calculating the location.

• Exercise 2

Code

```
edit ex2

tool pen

glue.empty = 1009
glue.ok = 1010

reacti glue.empty, recharge.gun

appro board:a, 50
moves board:a
closei
moves board:b
break
moves board:c
break
moves board:d
break
moves board:a
openi
departs 50
move wait.loc
set board = board:shift(null by 10, 10, 0)

e
```

```
edit recharge.gun

set current.pos=here
set current.dest=dest
openi
departs 50
appro recharge,50
moves recharge
wait sig(glue.ok)
departs 50
appro current.pos, 50
moves current.pos
closei
moves current.dest
reacti glue.empty, recharge.gun

e

load alltools.v2
load glue.lc
```

Questions

We handled the input asynchronously by the means of **reacti.** We chose to use that because we needed an immediate response and not to wait for the motion to finish. It's important to note that we had to reset the reacti at the end of itself, in order to be able to handle more than one interrupt per cycle. To make sure we glue properly even if an interrupt occurs is to restart the motion from the point in the trajectory where the motion stopped and to do so we need to save the current position in the current.pose variable.

We used this instruction set board = board:shift(null by 10, 10, 0) to apply a translation on the board frame.

- ## Exercise 3

Code

```
dit ex3

tool gripper

air.pressure = 5
signal air.pressure

set pallet.frame = frame (p4, p1, p2 ,p4)
shift.x= 82
shift.y= 59
obj.height=-15
row.max = 2
col.max = 3

for row=0 to row.max-1 step 1
  for col=0 to col.max-1 step 1
    set depose = depose:shift(null by 0,0,obj.height)
    set current.pose = pallet.frame:shift(null by col*shift.x,row*shift.y,0)
    appro current.pose, 50
    moves current.pose
    closei
    departs 50
    appro depose, 50
    moves depose
    openi
    departs 50
  end
end
move wait.loc
e

load pallet.lc
load alltools.v2
```

Questions

For this exercise we were given the coordinates of the 4 objects at the corners of the pallet. From those points we extract the pallet frame that will have origin in P4, the z axis will coincide with the tool frame z axis and the y axis will be aligned along the shorter side (thanks to the choice of point P2) to allow the correct functioning of the gripper.

We should consider the thickness of the object at the first time when we are putting it on the stack.

- ## Exercise 4-part-1

Code

```
edit ex4

tool null

speed 30 always
speed 100 mmps always

stop.condition = 1009
motion.step = 10
distance.point = 0
timer 1 = 0
arrival = 100
set init.point = here

do
set a.little.higher = shift(here by 0,0,motion.step)
moves a.little.higher
distance.point = distance(here,init.point)
until (sig(stop.condition) or (distance.point >= arrival))

time.taken = timer(1)
type "distance = ",distance.point
speed.average = distance.point/time.taken
type "speed = ", speed.average

e
```

Comments

We used a Do-Until loop with two conditions: the first to stop the motion if an input arrived and the second to stop the motion when arrived at the desired position. There was a problem with () while writing conditions for the loop.
The speed was really slow as we were checking the conditions each time before executing the new cycle.
We calculated the distance using distance(here, init.point), although that was not so accurate because we checked it synchronously. If a break instruction was added this problem would have been partially solved, but the motion would have stopped at every iteration, therefore we avoided it.

- Exercise 4-part-2

Code

; PUMA (VAL||)	; RX90 (V+)

edit ex4b
 	
tool null
 	
speed 30 always
speed 100 mmps always
 	
stop.condition = 1009
distance.point = 0
arrival = 400
set init.point = here
set a.little.higher = shift(here by 0,0,arrival)
pcexecute stop.check, -1
timer 1 = 0
moves a.little.higher
break
pcend
time.taken = timer(1)
speed.average =
distance(here,init.point)/time.taken
type "distance = ",distance(here,init.point)
type "speed = ", speed.average
 	
e
 	
edit stop.check
if sig(stop.condition) then
  brake
end
e
	edit ex4b
	
	tool null
	
	speed 30 always
	speed 100 mmps always
	
	stop.condition = 1009
	distance.point = 0
	arrival = 400
	set init.point = here
	set a.little.higher = shift(here by 0,0,arrival)
	execute 1 stop.check, -1
	timer 1 = 0
	moves a.little.higher
	break
	abort 1
	time.taken = timer(1)
	speed.average = distance(here,init.point)/time.taken
	type "distance = ",distance(here,init.point)
	type "speed = ", speed.average
	
	e
	
	edit stop.check
	if sig(stop.condition) then
	  brake
	end
	e

Comments

In this case we used two tasks: one to check the stop condition and one to make the robot's move. The robot would move faster this time because the task to check the condition was running in pseudo-parallel.