Ecole Centrale de Nantes

ARPRO IRP LAB Report

Xihao Wang and Ragesh Ramachandran

Exercise 1

1.1 exericise1 part1

Following the requirement of the title, we wrote the code and used this code to run on the Puma robot. And we choose to use of synchronous handling of inputs. After several modifications, we think this code is able to reach the requirement perfectly.

```
edit exercise1.program //(At the first step, we define the code’s name)
tool succion.pad //(Mentioned in the title, we use the suction pad as robot’s tool)
p1=1004 //(according to the robotics lab binary, the binary number of place 1 is 1004)
p2=1005 //(The binary number of place 2 is 1005)
e1=1007 //(The binary number of detector 1 is 1007)
e2=1008 //(The binary number of detector 2 is 1008)
signal(5) //(signal 5 is the air press controller’s switch, 5 means active the air press control)

wait sig(p1) or sig(p2) //(here is a select constructs which to meet the first requirement)
if sig(p1) then //(if the object is placed at the place 1)
appro p1, 50 //(the end effector approach the place where on the 50mm of place 1)
moves p1 //(The end effector directly moves to place 1 as a straight line)
closei //(Means end effector open the suction pad immediately to grasp the object)
depart 50 //( The end effector depart the place 1 and rise 50mm on the place 1)
appro place,100 //( the end effector approach the place where on the 100mm of the place,
place is the location which saved in the file and on the conveyor)
moves place //(The end effector directly moves to place as a straight line)
openi //(means end effector close the suction pad immediately to release the object on the
conveyor)
depart 100 //(The end effector depart the place and rise 100mm on the place)
move wait.loc //(The end effector move to wait.loc where the location also sved in the file)
else
appro p2,50 //(the situation of the place 2 is the same as the place 1)
moves p2
closei
departs 50
appro place,100
moves place
```

```
openi
departs 100
move wait.loc
end

signal 2,1 //(when the object be placed on the place, we open the conveyor. According to
robotics binary setting, number 2 means the direction from detector 1 to dtector2 and the
number 1 means open the conveyor)
wait sig(e2) //(wait the detector 2 detect the object)
signal -1 //
timer(1)=0 //(set a timer(1) and this timer equal to 0 now)
wait (timer(1)>2) //(wait 2 seconds, when then timer(1) goes to more than 2 seconds)
signal -2,1 //(after 2 seconds waiting, we start the conveyor again. But this time the direction
is from detector2 to detector1 so the number is -2)
wait sig(e1) //(waiting the detector 1 to detect the object)
signal -1 //(number 1 means stop the conveyor)

appro grasp.loc.stopped,100 //(The end effector approach to the place where on the 100mm
from the location named grasp.loc.stopped which also saved in the file)
moves grasp.loc.stopped //( The end effector directly moves to grasp.loc.stopped  as a
straight line)
closei //(Means end effector open the suction pad immediately to grasp the object)
depart 100 //(The end effector depart the grasp.loc.stopped and rise 100mm on the location
grasp.loc.stopped)
appro p3,100 //( the end effector approach the place where on the 100mm of the place 3)
moves p3 //( The end effector directly moves to place 3 as a straight line)
openi //(means end effector close the suction pad immediately to release the object on the
place 3)
depart 100 //(The end effector depart the place 3 and rise 100mm on the location place 3)
move wait.loc //( The end effector moves to wait.loc)
signal(-5) //(-5 means shut down the air press switch)
e

load pickconv.lc //(those 2 file saved the position and other thing that will be used in the
program)
load alltools.v2
//(here has a space)
```

1.2 exercise1 part2

Before the code, there are 3 questions to answer:

a. In the second part of the exercise, why should you let the part pass the second detector before changing the direction of the conveyor?

   A: This is because we set a timer which in order to count how long the object spend from detector2 to detector1. If we want to get the precise time, we need to let the object go through the detector2 at the time point equal to 0.

Therefore, we have to let object pass the detector2 at first and then change the direction to let the object pass detector2 again when the timer is at the time point equal to 0 rather than just let the object stop at the location which the same as the detector2.

b. Test the program at low/high speed. What are the reasons for your program to fail?

A: In our program, it performs well in low speed and the end effector is able to grasp the object successful in low speed. However, in high speed, the end effector can’t grasp the object. But I think is the reason is too fast speed which lead the suction pad have not enough time to grasp the object.

c. In the location file « pickconv.lc », you have the location *grasp.loc.stopped* for grasping the part. How would you operate to teach this location to the robot?

A: we have to set a new location named *new.grasp.loc.stopped.* We can calculate this location. At the first step, we get the time which the object spends from detector2 to detector1 and the distance from detector2 to detector1 is 700mm. Then, we are able to use 700/time to get the speed of conveyor. According to the title, there has a triangle relationship from conveyor distance and end effector move distance. So we can calculate that the end effector speed equal to (700/time)*sqrt(2) and the location *new.grasp.loc.stopped* is 150mm behind the location *grasp.loc.stopped* .

Following the requirement of the title, we wrote the code and used this code to run on the Puma robot. And we choose to use of synchronous handling of inputs. After several modifications, we think this code is able to reach the requirement perfectly. We will not commend the code which has appearance in the part1.

```
edit e1v2
speed 30 always //(as the requirement in the title)
speed 100mmps always
tool succion.pad
det.p1=1004
det.p2=1005
e1=1007
e2=1008
signal(5)

wait sig(det.p1) or sig(det.p2)
if sig(det.p1) then
set grab = p1
else
set grab = p2
```

```
end
appro grab,50
moves grab
closei
departs 50
appro place,100
moves place
openi
departs 100
appro grasp.loc.stopped,150

signal 2,1
wait sig(e2)
timer(1)=0
wait (timer(1)>3) //(When the object passed the detector2, the timer start to count second. Therefore, the object will still move forward in following 3s)
signal -1 //(After the object have passed the detector2 for 3 seconds, the conveyor stop)
timer(1)=0 //(Setting a new timer which start from 0 again
wait (timer(1)>2) // The purpose of this timer is let the object to stop in 2 seconds)
signal -2,1 //(After the object stoped for 2 seconds, the conveyor start work again, but work on the opposite direction)
wait sig(e2)
timer(1)=0 //When the object be detected by detector2, we setting a new timer which start from 0)
wait sig(e1)
t=timer(1) //(When the object have been detected by detector1, t equal to the time that the object spend from detector2 to detector1)
v=700/t //(In this way, we can calculate the speed of the conveyor)
v.effector=v*sqrt(2) //(According the relationship mentioned in the title, we are able to get the end effector's speed)
speed v.effector mmps //(Setting the end effector's speed)
set new.grasp.loc.stopped = shift(grasp.loc.stopped by 0,150,0) //(Because this is a isosceles triangle, so the distance between previous and new grasp location is equal to the height the end effector waiting location which is 150mm above the previous grasp location. And using the shift sentence to set the new grasp location after previous location in the y direction)
moves new.grasp.loc.stopped //(Because the result have been calculated, so the end effector will grasp the object at the new grasp location even though the object still moving)
closei
depart 100
appro p3,100
moves p3
openi
depart 100
move wait.loc
signal(-5)
e

load pickconv.lc
load alltools.v2
```

Exercise2

Before the code, there are 3 questions to answer:

a. Should the glue default event be handled synchronously or asynchronously?
   A: The glue default event should be handled asynchronously.

b. Which are the Val/V+ instructions that can be used? Which one will you choose and why?
   A: The instructions “reacti”, “appro”, “moves”, “depart”, “break”, “set”, are used in our code.
   For the “reacti”, this is because that if the input9 detects the glue gun is empty, the glue pen should be stop immediately. We need a instruction to react with this situation, so we choose “reacti”.
   And other movement instructions are used in many program.
   Moreover, the reason to use instruction “break” is because that if we not use this instruction, the trajectory will not follow the correct rectangle. The trajectory will draw a error shape which the rectangle have four round corners and those four points became the fly-by points.

c. What is the solution to make sure that the object for which a glue default occurred is correctly glued along the whole trajectory?

   A: Add the sentence “set board = board:shift(null by 10,10,0) ” in the code. This is because that *A*, *B*, *C* and *D* are given in “glue.lc” as locations relative to the *board* frame. At the end of the program, shift locations *A*-*D* 10 mm along X and Y. So we need to add a transform on board, just like add a matrix, to correct those 4 points location and in order to execute the program in infinite loop. Otherwise, the trajectory will gradually deviate the correct way.

Following the requirement of the title, we wrote the code and used this code to run on the RX90 robot. And we choose to use of asynchronous handling of inputs. After several modifications, we think this code is able to meet the requirements perfectly.

```
edit exercise2.program
tool pen //(this time, our tool is the glue pen)
appro board:a,50 //(the location of point a, b, c and d have been saved in the file, so the glue
pen move to the a point above 50mm)
move board:a
closei //(when the glue gun move to the point a, closei means it will start gluing
immediately)
```

```
reacti 1009,gluefill //(the robot make a judgement depending on the detect result from
input9, if the glue gun is filled, the robot will run the following code. If the glue gun is
empty, the robot will stop drawing and running code “gluefill” which wrote at behind.)
moves board:b
break //(the reason why we used “break” have been answered in the question b
moves board:c
break
moves board:d
break
moves board:a
break
openi //(after the glue pen draw a complete rectangle, it stop gluing immediately)
set board = board:shift(null by 10,10,0) //( the reason why we wrote this sentence to
transform the board’s position we have been answered in the question c)
depart 50
move wait.loc
e
//(the normal situation part code is over, the following code is meet the situation when the
glue gun is empty)
edit gluefill
set current.loc=here //(we set a position named “here” which marked the stop point in order
to help the glue gun back to its position successfully)
set arrival=dest //(we record robot’s current destination which means the next point that the
glue gun will approach.)
Openi //(even though the glue gun is empty, anyway, we close the glue gun)
appro recharge,50 //(The glue gun will move to the place that recharge the glue gun)
moves recharge
wait sig(1010) //( we set 10 seconds for glue gun to complete its recharge work)
departs 50
appro current.loc,50 //(after the glue gun filled again, it will find the stop position where we
marked it before.)
moves current.loc
closei //(when the glue gun back to the stop point, it start to draw again)
moves arrival //(we have recorded the next point, so the glue gun will follow it previous
trajectory to continue its work)
e

load glue.lc
load alltools.v2
```

Exercise 3

Following the requirement of the title, we wrote the code and used this code to run on the RX90 robot. And we choose to use of synchronous handling of inputs. After several modifications, we think this code is able to reach the requirement perfectly.

```
edit ex3
```

```
tool gripper
open.valve = 5 //(according to the binary of the robot, the air pressure switch also is 5)
signal(open.valve)
set point.frame = frame(p4,p1,p2,p4) //(we use three points to define a frame, the points are p4, p1 and p2 which are have already saved in the file and p4 is the original point)
x = 82 //(we have knew that the distance between two objects in x direction is 82mm)
y = 59 //(we have knew that the distance between two objects in y direction is 59mm)
t = 15 //(we have knew that the thickness of each object is 15mm)
i.pal = 2
j.pal = 3
for i.pal=0 to i.pal-1 //(there are two cycles, which let the end effector grasps the objects one by one and following a order)
for j.pal=0 to j.pal-1
set cur.loc = point.frame:shift(null by j.pal*x,i.pal*y,0) //(according to the position shown in title, we set the cur,loc to represent the position of object that will be grasped at this time. And the movement that to find next object is depending on the distance between two objects in both x and y direction)
appro cur.loc, 50
moves cur.loc
closei //(The end effector grasps the object)
departs 50 //(The end effector must rise a distance before the next movement, otherwise the end effector will crash the platform)
set depose = shift(depose by 0,0,t) //(we set a depose location and this location will rise 15mm each time according to the thickness of the object. Therefore, each time the grasped object will smoothly place on the previous object)
appro depose, 50
moves depose
openi //(The end effector releases the object)
departs 50
end
end
move wait.loc
e

load pallet.lc
load alltools.v2
```

Exercise4

Before the code, there are 2 questions to answer:

a. Compared with guarded motion and average motion, which solution is best?

A: We think the guarded motion is better because the guarded motion is able to protect the robot when robot have to stop emergency. If let the robot constantly moving at desire speed, it will spend much longer time to stop and deviate the previous position at a long distance.

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

> **A:** Because the guarded motion is composed by many small distances and it will experienced many time when it speed equal to 0. Therefore, it will spend much longer time to reach the final position compared with constantly moving at a desired speed. So it average speed always lower than the desired speed.

Following the requirements of the title, we wrote the code and used this code to run on the RX90 robot. This exercise is composed by 2 part of code.
At the first part, this motion is designed as the guarded motion which

```
edit program4
speed 30 always
speed 100 mmps always
tool null //(This time we don’t use any tool)
set init.pos = here //(Initial position is controlled by using keyboard)
set final.pos = shift(init.pos by 0, 0, 400) //(The final position is 400mm away from the initial position)
set move.pos = shift(init.pos by 0, 0, 2) //(The move position means the distance at the each movement in the guarded motion, the each step at the motion is 2mm which is small enough compared with the whole distance between initial position and final position)
timer 1 = 0 //(The motion start at the timer equal to 0)
do //(here is a loop until the end effector reach the final position)
moves move.pos
break //(The end effector move 2mm will stop and start again. In this way, a straight line motion is displaced into many small pieces of movement)
set move.pos = shift(move.pos by 0, 0, 2)
until ( (distance(move.pos, final.pos)<2) or sig(1009) ) //(when the distance between current location and the final position is less than 2mm or input 1009 button be pushed, the end effector stop means the motion finished)
dist = distance(here, init.pos)
sp1 = dist/timer(1) //(use distance/time to calculate the average speed of the guarded motion in order to compare with desired speed)
type "avg speed=",sp1
e
```

Second part is the motion followed the desired speed to reach the final location. If push the input 1009 button, the motion will stop immediately.

```
edit program4
speed 30 always
speed 100 mmps always
tool null
set init.pos = here
set final.pos = shift(init.pos by 0, 0, 400)
timer 1 = 0
pcexecute monitoring, -1 //(Execute n iterations of program. Because here the number is -1, so the program loops infinitely.)
```

```
moves final.pos //(this time the end effector directly move to final position without any stop
or break)
pcend //(Stop program at the end of current iteration.)
break
dist = distance(here, init.pos)
sp1 = dist/timer(1)
type "avg speed=",sp1
e

edit monitoring
if ((distance(here, final.pos)< 2) or sig(1009)) then
brake
end
e
```