THE UNIVERSITY OF MICHIGAN-DEARBORN
SCHOOL OF ENGINEERING
INDUSTRIAL AND MANUFACTURING SYSTEMS ENGINEERING DEPARTMENT

IMSE/CIS 381: INDUSTRIAL ROBOTICS

ASSIGNMENT #4
Robot Programming

Student Name: Demetrius Johnson
Date: March 20, 2021

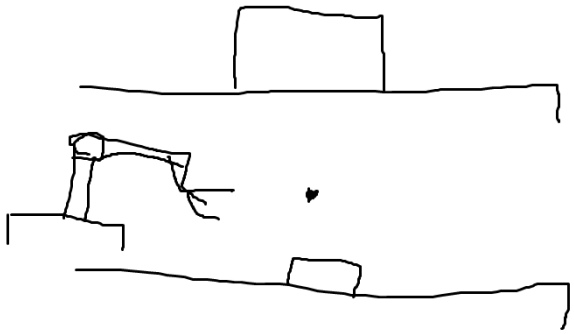
1) Program the robot to pick up two blocks (the blocks are different sizes) from fixed positions on either side of a center position, and stack the blocks in the center position. The larger block will always be on one side of the center and the smaller block will always be on the other side of the center position. The smaller block is to be placed on top of the larger block.

PROGRAM DEFINITIONS (GLOBAL)

- 1: DEFINE point C_L //center point for Large block placement at the center point
- 2: DEFINE point C_S //center point for Small block placement atop Large block
- 3: DEFINE point H // home position 100mm above point C_L
- 4: DEFINE point L_H // (High) above center of Large block
- 5: DEFINE point L_G // grab position for the Large block
- 6: DEFINE point L_H //100mm (High) above center of Large block
- 7: DEFINE point L_G // grab position for the Large block
- 8: DEFINE point S_H // (High) above center of Small block
- 9: DEFINE point S_G // grab position for the Small block
- 10: DEFINE line 1: open Gripper //output line to open gripper
- 11: DEFINE line 2: close Gripper //output line to close gripper
- 12: DEFINE line 3: move Conveyors //moves the conveyors of both large and small blocks in order to obtain a new set of blocks

PROGRAM SUBROUTINES (MACROS/BRANCHES/FUNCTIONS)

- 13: BRANCH GET_LARGE
- 14: MOVE L_H //move to Large home position
- 15: MOVE L_G //move to Large grab position
- 16: SIGNAL 2 //close gripper
- 17: MOVE L_H //move back to Large home position
- 18: MOVE H //move to home location
- 19: MOVE C_L //move to center location
- 20: SIGNAL 1 //open gripper; release Large block
- 21: END GET_LARGE

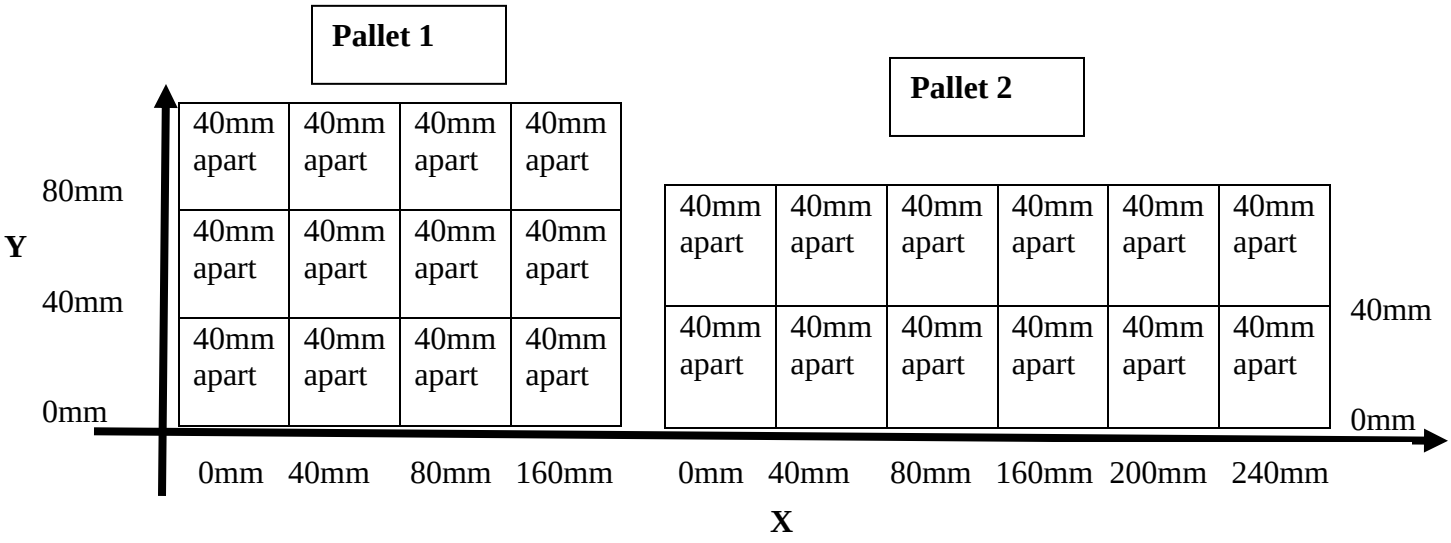


- 22: BRANCH GET_SMALL
- 23: MOVE S_H //move to Small home position
- 24: MOVE S_G //move to Small grab position
- 25: SIGNAL 2 //close gripper
- 26: MOVE S_H //move back to Small home position
- 27: MOVE H //move to home location
- 28: MOVE C_S //move to center location for Small block
- 29: SIGNAL 1 //open gripper; release Small block
- 30: END GET_SMALL

MAIN FUNCTION

- 31: MOVE H //start at home position
- 32: SIGNAL 1 //open gripper
- 33: GET_LARGE //subroutine to get large block and stack it at center
- 34: MOVE H
- 35: GET_SMALL //subroutine; get small block and stack it at center atop large one
- MOVE H
- 36: SIGNAL 3 //move conveyors to bring forward the next blocks
- 37: GO TO 31 //repeat the main function
- 38: END PROGRAM

2) A robot is to be programmed to unload parts from one pallet and load them onto another pallet. The parts are located on the unload pallet (pallet 1) in a 3 by 4 pattern in known fixed positions, 40 mm apart in both directions. The two directions of the pallet are assumed to be parallel to the x and y world coordinate axes of the robot. The parts are to be placed on the load pallet (pallet 2) in a 2 by 6 pattern, 40 mm apart in both directions. The two directions of the pallet are again assumed to be parallel to the x and y world coordinate axes of the robot. Make a sketch of the workstation setup before you begin programming.



PROGRAM DEFINITIONS (GLOBAL)

1. DEFINE point P1_CORNER = [JOINT COORD] //define center for bottom-left corner of pallet 1
2. DEFINE point P2_CORNER = [JOINT COORD] //define center for bottom-left corner of pallet 2
3. DEFINE INTEGER X_P1 //x-coordinate value
4. DEFINE INTEGER Y_P1 //y-coordinate value
5. DEFINE INTEGER X_P2 //x-coordinate value
6. DEFINE INTEGER Y_P2 //y-coordinate value
7. DEFINE PICKUP_P1 = COORDINATES (X, Y)
8. DEFINE DROP_P2 = COORDIANTES (X, Y)
9. DEFINE int ROW_P1
10. DEFINE int COL_P1
11. DEFINE int ROW_P2
12. DEFINE int COL_P2

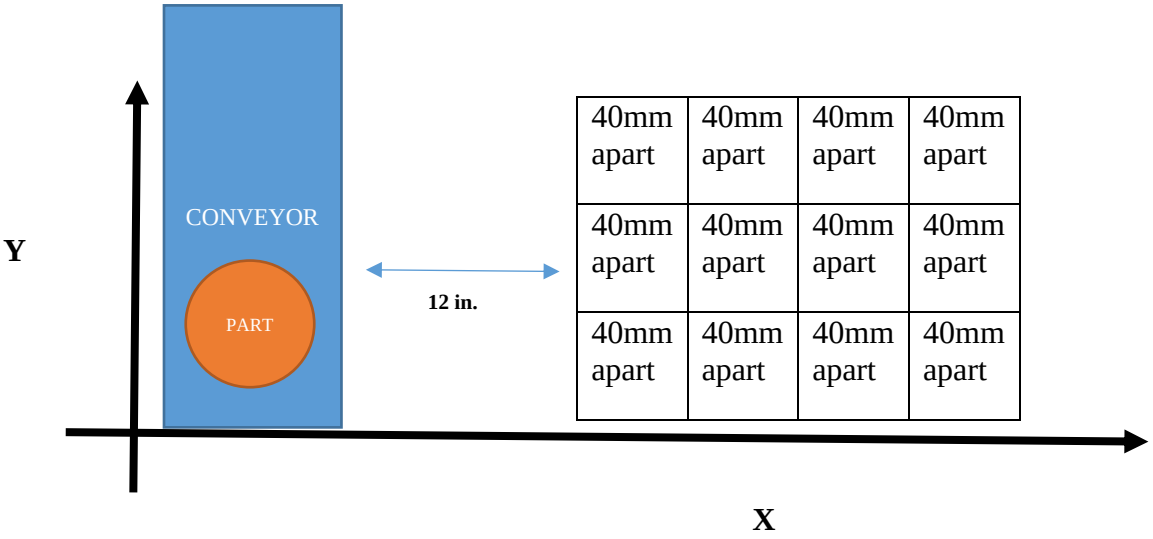
PROGRAM SUBROUTINES (MACROS/BRANCHES/FUNCTIONS)

13. **BRANCH REINITIALIZE_ROW_COL**
14. IF COL_P1 == 3
15. THEN:
    a. COL_P1 = 0
    b. ROW_P1 = ROW_P1 + 1 //after reinitializing a column, it means we need to go to the next row
16. ELSE:
    a. COL_P1 = COL_P1 + 1 //only increase columns if it is not out of range
17. END IF
18. IF COL_P2 == 5
19. THEN:
    a. COL_P2 = 0
    b. ROW_P2 = ROW_P2 + 1 //after reinitializing a column, it means we need to go to the next row
20. ELSE:
    a. COL_P2 = COL_P2 + 1 //only increase columns if it is not out of range
21. END IF
22. END REINITIALIZE_ROW_COL
23. **BRANCH SET_COORDINATES**
24. X_P1 = COL_P1 * 40
25. X_P2 = COL_P2 * 40 //multiply the X and Y coordinate values by the spacing == 40mm
26. Y_P1 = ROW_P1 * 40
27. Y_P2 = ROW_P2 * 40
28. END SET_COORDINATES

MAIN FUNCTION

29. OPENI //start with gripper open
30. ROW_P1 = 0
31. ROW_P2 = 0 //initialize all rows to 0 at start of program
32. COL_P1 = -1 //initialize columns to -1 so that they will go into the branch statement and be set to 0
33. COL_P2 = -1
34. REINITIALIZE_ROW_COL //increment rows or columns as necessary, reset columns to 0 if necessary
35. IF (ROW_P1 == 3 AND ROW_P2 == 2) GO TO 48 //if rows and columns out of scope; end program
36. SET_COORDINATES //set coordinate values based on ROW and COL and 40mm spacing
37. PICKUP_P1 = P1_CORNER + <X_P1, Y_P1> //offset from pickup corner on pallet 1
38. APPROS PICKUP_P1 50 //approach pickup point from 50mm above
39. MOVE PICKUP_P1
40. CLOSEI
41. DEPART 50
42. DROP_P2 = P2_CORNER + <X_P2, Y_P2> //offset from drop-off corner on pallet 2
43. APPROS DROP_P2 50
44. MOVE DROP_P2
45. OPENI
46. DEPART 50
47. GO TO 34
48. END PROGRAM

3) Write a robot program to pick parts off a conveyor and load them into a pallet that is about 12 in. from the pickup point. A mechanical stop on the conveyor is used to locate the parts in a known position for the pickup. The parts are to be arranged in a 3 by 4 pattern, 40 mm apart in both directions. The two directions of the pallet are assumed to be parallel to the x and y world coordinate axes of the robot, respectively.



40mm apart	40mm apart	40mm apart	40mm apart
40mm apart	40mm apart	40mm apart	40mm apart
40mm apart	40mm apart	40mm apart	40mm apart

PROGRAM DEFINITIONS (GLOBAL)

1. DEFINE point PALLET_CORNER = [JOINT COORD] //define center for bottom-left corner of pallet
2. DEFINE point PART_LOCATION = [JOINT COORD] //define center of part location on the conveyor
3. DEFINE INTEGER X_PALLET //x-coordinate value
4. DEFINE INTEGER Y_PALLET //y-coordinate value
5. DEFINE DROP_PALLET = COORDIANTES (X, Y)
6. DEFINE int ROW_PALLET
7. DEFINE int COL_PALLET

PROGRAM SUBROUTINES (MACROS/BRANCHES/FUNCTIONS)

8. **BRANCH REINITIALIZE_ROW_COL**
9. IF COL_PALLET == 3
10. THEN:
    a. COL_P1 = 0
    b. ROW_P1 = ROW_P1 + 1 //after reinitializing a column, it means we need to go to the next row
11. ELSE:
    a. COL_P1 = COL_P1 + 1 //only increase columns if it is not out of range
12. END IF
13. END REINITIALIZE_ROW_COL
14. **BRANCH SET_COORDINATES**
15. X_PALLET = COL_PALLET * 40
16. Y_PALLET = ROW_PALLET * 40 //multiply the X and Y coordinate values by the spacing == 40mm
17. END SET_COORDINATES

MAIN FUNCTION

18. OPENI //start with gripper open
19. ROW_PALLET = 0 //initialize all rows to 0 at start of program
20. COL_PALLET = -1 //initialize columns to -1 so that they will go into the branch statement and be set to 0
21. REINITIALIZE_ROW_COL //increment rows or columns as necessary, reset columns to 0 if necessary
22. IF (ROW_P1 == 3) GO TO 34 //if rows and columns out of scope; end program
23. SET_COORDINATES //set coordinate values based on ROW and COL and 40mm spacing
24. APPROS PART_LOCATION 50 //approach pickup point from 50mm above
25. MOVE PART_LOCATION
26. CLOSEI //close gripper
27. DEPART 50
28. DROP_PALLET = PALLET_CORNER + <X_PALLET, Y_PALLET> //offset from drop-off corner on pallet 2
29. APPROS DROP_PALLET 50
30. MOVE DROP_PALLET
31. OPENI
32. DEPART 50
33. GO TO 21
34. END PROGRAM

4) Write a simple program to move the robot through a path that consists of 15 points along the line Y= 0.5X.

PROGRAM DEFINITIONS (GLOBAL)

1. DEFINE INTEGER X
2. DEFINE INTEGER Y
3. DEFINE INTEGER COUNTER
4. DEFINE point INITIAL = [JOINT COORD] //use this to record an initial starting point
5. DEFINE point LINEAR = COORDINATES (X, Y) //use this to store the XY coordinates of a point

PROGRAM SUBROUTINES (MACROS/BRANCHES/FUNCTIONS)

6. BRANCH SET_XY
7. X = X + 1
8. Y = 0.5 * X //set Y based on X
9. END SET_XY

MAIN FUNCTION

10. COUNTER = 0
11. X = -1 //initialize X at -1 since first branch call will set it to 0
12. SET_XY //this branch will initialize Y
13. LINEAR = INITIAL + <X, Y> //set new position for LINEAR variable
14. MOVES LINEAR //move in a straight line to location stored in LINEAR
15. COUNTER = COUNTER + 1 //now increment counter
16. IF (COUNTER < 15) GO TO 12 //set X and Y to new values if COUNTER < 15
17. END PROGRAM

5) An industrial robot performs a machine loading and unloading operation. A PLC is used as the robot cell controller. The cell operates as follows:

- -(1) a human worker places a work part into a nest,
- -(2) the robot reaches over and picks up the part and places it into an induction heating coil,
- -(3) a time of 10 seconds is allowed for the heating operation, and
- -(4) the robot reaches in and retrieves the part and places it on an outgoing conveyor.
- A limit switch I1 (normally open) will be used in the nest to indicate part presence in step (1).
- Similarly the presence of a part on the induction heating coil in step (2) and the outgoing conveyor in step (4) will be detected by I2 and I3, respectively.
- Output contact O1 will be used to signal the robot to execute step (2) of the work cycle. This is an output contact for the PLC, but an input interlock for the robot controller.
- Timer T1 will be used to provide the 10 second delay in step (3).
- Output contact O2 will be used to signal the robot to execute step (4).

Construct the ladder logic diagram for the system.

- I1 = detect part presence in the nest
- I2 = detect presence of a part in the heating coil
- I3 = detect presence of part on conveyor
- O1 = signal robot to execute pick up part from nest and place into heating coil (This is an output contact for the PLC, but an input interlock for the robot controller.)
- O2 = signal robot to execute retrieve part from heating coil and place it on conveyor
- T1 = 10 second time delay in heating coil from step 3 (wait)