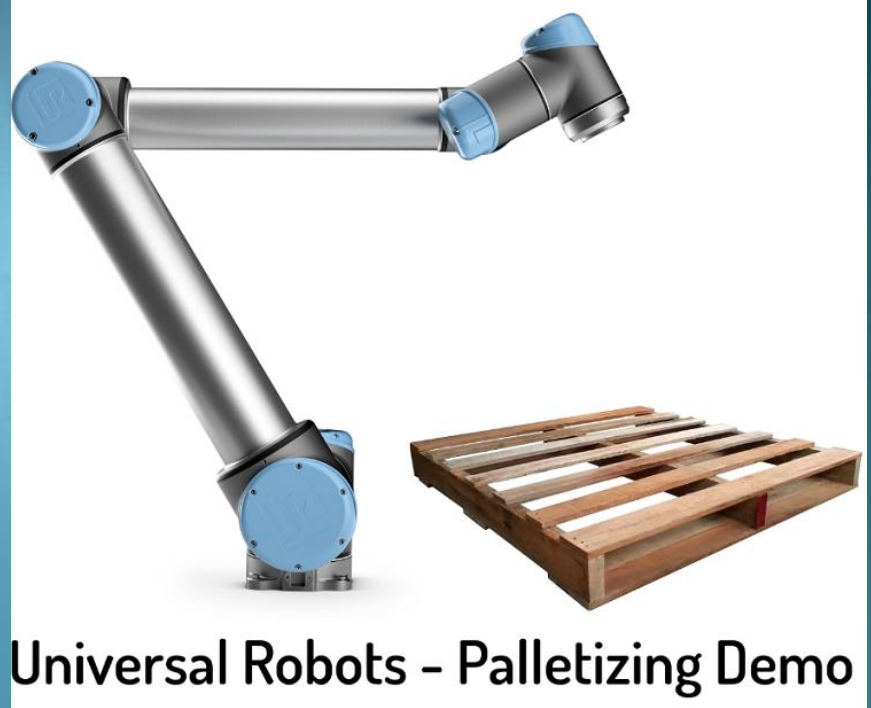


Welcome to Universal Robots Online Workshop on Palletizing

Trainer-
Maneesh Garg
Tech Support India



Universal Robots - Palletizing Demo



Simple Palletizing

- Pallet Wizard will help you when the orientation of Boxes are same through out the line, plane and Box.
- Some time many number of pallet wizards are required to teach the box positions depending upon their orientations/line or layer.



Pallet wizards

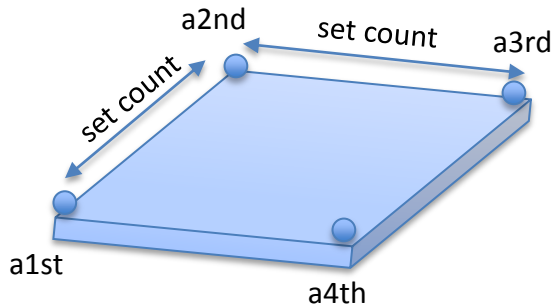
- Pattern
 - Determine palletizing pattern
 - Patterns
 - Line
 - Square
 - Box
 - List
- PalletSequence
 - What to do at each position in pattern

The screenshot displays the Universal Robots software interface. The top menu bar includes 'File', 'Program', 'Installation', 'Move', 'I/O', and 'Log'. The 'Program' tab is active, showing a tree view on the left with 'Robot Program' expanded to 'Pallet', then 'Pattern', 'PalletSequence', 'Approach', 'PatternPoint', 'Set', 'Wait', and 'Exit'. The main window is titled 'Pattern' and contains the following text: 'A pattern is a group of positions to be cycled through. Patterns can be used for making much more palletizing etc.' Below this text are four options, each with a button and a diagram: 'Line' (a dashed line with two points), 'Square' (a dashed square with four points, highlighted with a blue border), 'Box' (a dashed 3D box with eight points), and 'List' (a cluster of ten dashed points). At the bottom of the interface, there are simulation controls: 'Simulation' (selected), 'Real Robot', and a 'Speed' slider set to 100%.

Pattern: Square

- Pattern
 - Set pattern: Square
 - Set objects between
 - Point 1 to 2
 - Point 2 to 3
 - Teach the 4 corners

Pattern



The screenshot displays the Universal Robots software interface for configuring a 'Pattern: Square'. The left sidebar shows a tree view under 'Robot Program' with 'Pallet' expanded to 'Pattern: Square', listing sub-items: 'a1st_Corner', 'a2nd_Corner', 'a3rd_Corner', 'a4th_Corner', 'PalletSequence', 'Approach', 'PatternPoint', 'Set', 'Wait', and 'Exit'. The main panel is titled 'Pattern: Square' and includes a 'Change type' button. Below the title, it states 'The counting variable 'cnt 1' used for traversing' and has a checkbox for 'Remember traversal position between program runs'. Two input fields are shown: 'Point 1 to 2 interval count' and 'Point 2 to 3 interval count', both with the value '5' entered. A 'Shared Parameters' section at the bottom right shows 'Tool Speed' at 250 mm/s and 'Tool Acceleration' at 1200 mm/s². At the bottom of the interface, there are buttons for 'Add new point' and 'Remove point', and a speed control slider set to 100%.

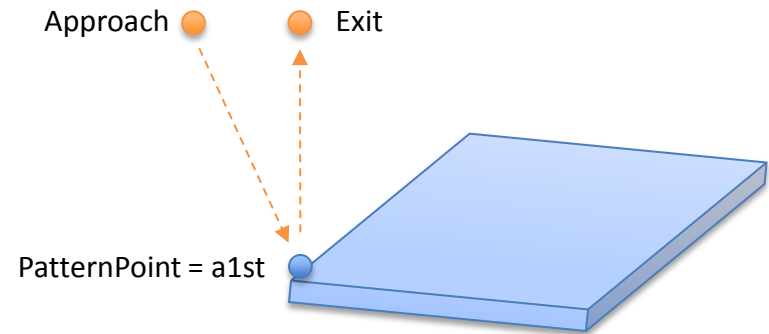
Core Training
Wrap-up

PalletSequence

- PalletSequence
 - Teach points
 - PatternPoint
 - Approach
 - Exit
 - Define actions

```
Robot Program
Pallet
  Pattern: Square
  a1st_Corner
  a2nd_Corner
  a3rd_Corner
  a4th_Corner
  PalletSequence
  Approach_1
  PatternPoint_1
  Set DO[0] =True
  Wait 0.5
  Exit_1
```

PalletSequence



- Rule of thumb: teach PatternPoint as some points as a1st_Corner
- Save sample program as pallet.urp

Cross Box Palletizing

- The way we generally follow to teach the boxes for a uneven pallet is all box teaching, which is tedious task and time consuming, even it will require frequent re-teachings in case of any change.
- The best way to teach this type of pallets is to use position offsets depending upon box dimensions.
- Benefit- No need to do teaching at Customer place, Very flexible, Universal program for all palletizing applications, Just do the mathematical changes and upload the working program.



In this Workshop we will use below programming features:

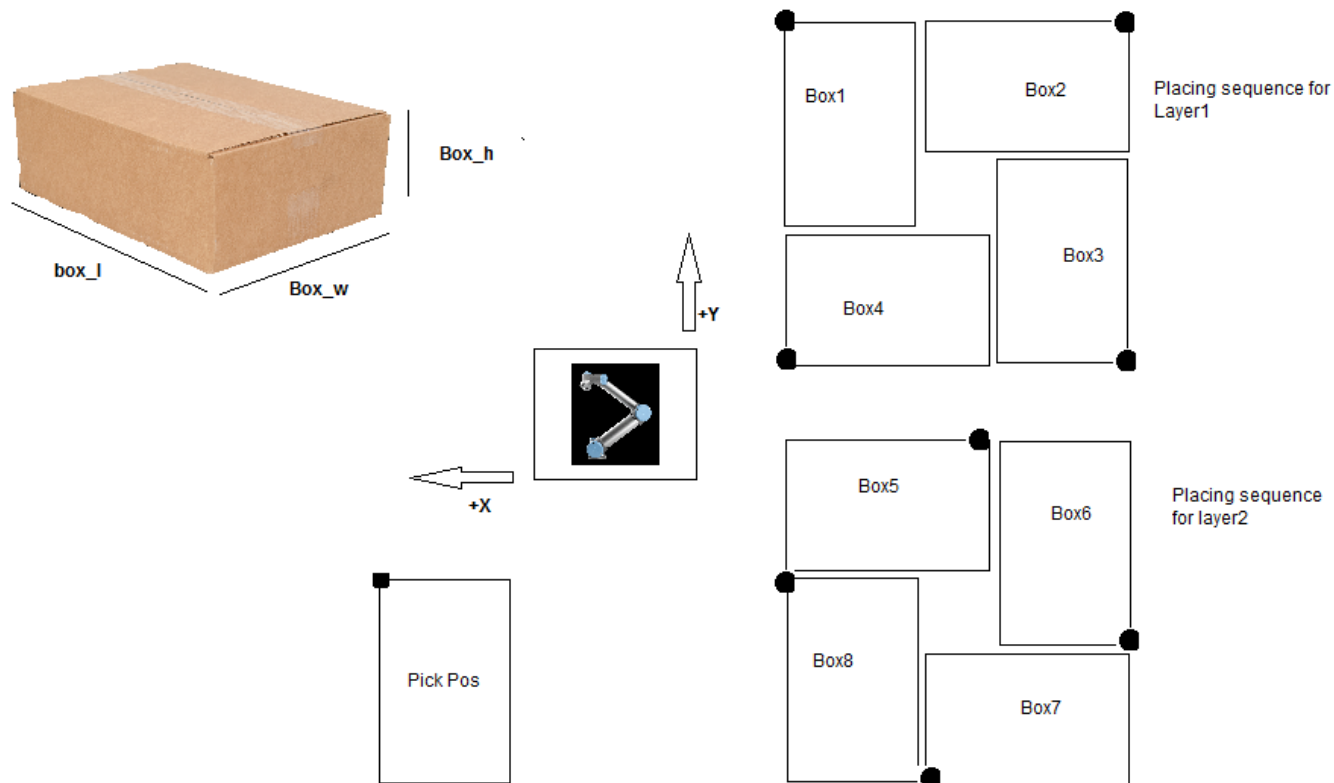
- Only 2 waypoints teaching is required
 - Pick reference position
 - Place reference position
- All other positions will be calculated in offsets.
- All offsets will be calculated in terms of box dimensions, if we change the box dimensions entire pallet will shift automatically and should be able to adjust all positions proportionally.
- No need to do teaching for Left & right pallet individually. Program itself should take care of both the pallets.

Description of Variables used:

1. **dd: drop distance- distance between actual placing position and ppre placing position.**
2. **Box_l, Box_w, Box_h: Dimensions of Box in meter**
3. **Gap: distance to be maintained between two consecutive boxes in meter**
4. **Box1,2,3,4,5,6,7,8: Variable positions of Boxes pre offsetted with required position and orientation.**
5. **Box Number: Number of boxes placed**

In next slide we will take an example of cross orientation of Pallet for 2 different layers and do the calculations for positioning.

Example :



- These Points need to be considered as teaching points while calculating the offsets

Calculation of variable position for all the boxes:

```
box1:=Place_ref
box1_up:=pose_trans(box1,dd)
box2:=pose_trans(box1,p[-(box_l+box_w+gap),0,0,0,0,d2r(90)])
box2_up:=pose_trans(box2,dd)
box3:=pose_trans(box2,p[0,-(box_l+box_w+gap),0,0,0,d2r(90)])
box3_up:=pose_trans(box3,dd)
box4:=pose_trans(box3,p[(box_l+box_w+gap),0,0,0,0,d2r(90)])
box4_up:=pose_trans(box4,dd)
box5:=pose_trans(box1,p[-(box_l-box_w),0,-box_h,0,0,d2r(90)])
box5_up:=pose_trans(box5,dd)
box6:=pose_trans(box2,p[0,-(box_l-box_w),-box_h,0,0,d2r(90)])
box6_up:=pose_trans(box6,dd)
box7:=pose_trans(box3,p[(box_l-box_w),0,-box_h,0,0,d2r(90)])
box7_up:=pose_trans(box7,dd)
box8:=pose_trans(box4,p[0,(box_l-box_w),-box_h,0,0,d2r(90)])
box8_up:=pose_trans(box8,dd)
```

URSim3.4.0.39 - VMware Workstation 12 Player (Non-commercial use only)

Player ▾ | [Icons]

[Icons]

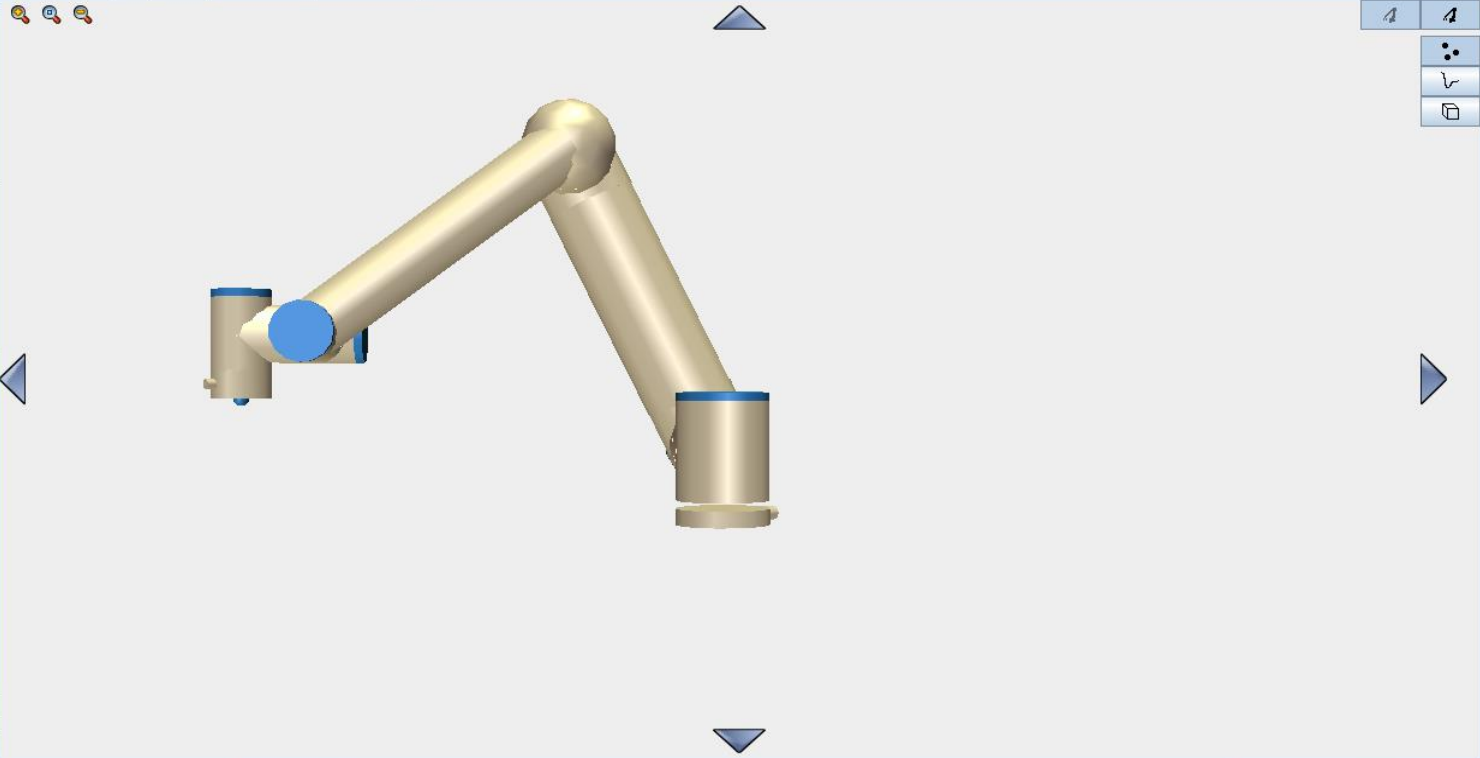
File 11:10:10 CCCC

Program Installation Move I/O Log

Palletizing made easy

Command Graphics Structure Variables

```
▼ BeforeStart
  ▼ MoveL
    = dd:=p[0,0,-0.12,0,0,0]
    = box_l:=0.2
    = box_w:=0.1
    = box_h:=0.1
    = gap:=0.01
  ▼ Calculation-Do not modify
    = box_no:=Enter the number of Placed boxes
  ▼ Robot Program
    'To be used only while teaching- While running
    ▼ pick
      = pick_up:=pose_trans(pick,dd)
      ▼ MoveL
        • pick_up
        • pick
        = Wait: 0.5
        • pick_up
    ▼ place
      ▼ If box_no≠0
        ▼ MoveL
          • box1_up
          • box1
          = Wait: 0.5
          • box1_up
          = box_no:=box_no+1
        ▼ Elself box_no≠1
          ▼ MoveL
            • box2_up
            • box2
            = Wait: 0.5
            • box2_up
            = box_no:=box_no+1
          ▼ Elself box_no≠2
            ▼ MoveL
              • box3_up
              • box3
              = Wait: 0.5
```



Simulation Real Robot

Speed 100%

Previous Next



- **Sample Program is attached.**

Lab Exercise:

Create the modifications in the attached program to teach different matrix & different box sizes.

Thank You

