Logic Challenge – Solution

-by Sandeep Patil

Problem Statement:

During the operation of picking up boxes from the pallet using a robot, decide the order of picking of boxes based on collision avoidance.

Given:

1. Position of Robot (X, Y, Z) = (0,0,0)
2. Height of tallest item = 400mm
3. All objects are right parallelogram prism
4. Object (box) properties:
    a. Center of the object = [object.center.x, object.center.y, object.center.z]
    b. X dimension = [object.x]
    c. Y dimension = [object.y]
    d. Maximum Z dimension: [object.z] ≤ 400mm
5. There is no restriction of number of boxes stacked.
6. Maximum height to which the robot can pick the object = 400mm from the object's initial position.
7. The dimensions of the flat pallet (width, breadth, height) = (1000mm, 1000mm, 0mm)
8. Items can slide on top of each other.
9. Robot will go on top of the box at a height = 100mm from surface.

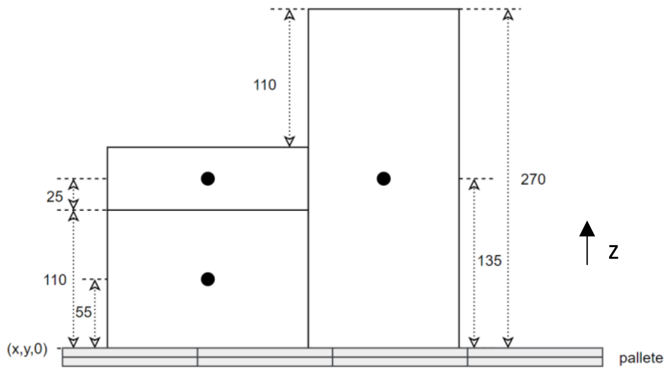
Assumptions:

1. Object centers [object.center.x, object.center.y, object.center.z] in mm are centers in the global coordinates (with (0,0,0) at the robot's center).
2. Vacuum gripper is used to pick up the boxes.
3. There is no collision/interference between robot and other boxes during the gripping action.
4. Boxes are assumed to be parallel to the coordinate axis.

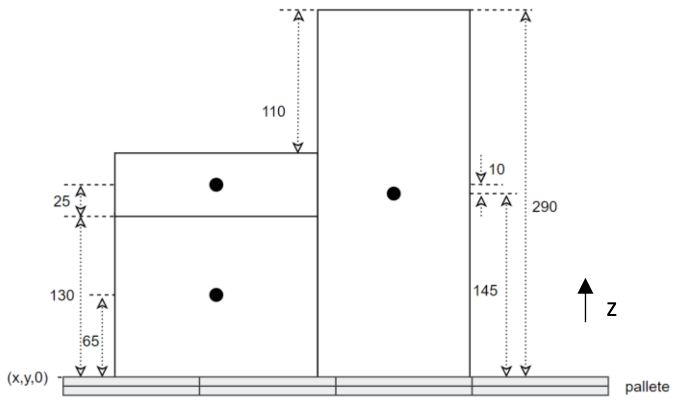
Methodology:

A simple solution could be given as first object in the sorted list of object.center.z of the objects. But there are two main edge case during which picking box with highest object.center.z will cause collision between the robot and the box.

1. Consider three boxes as shown in the Figure below, the tallest object (right) and the next object (left) have at the same centers. But the length between the two objects' top surfaces is 110mm which is greater than the height at which the robot traverses. This causes a collision between the robot and the topmost box.



2. Consider three boxes as shown in the figure below, the tallest object's (right) center is lower than the next object (left) by 10mm. Here picking the object which higher object.center.z causes collision with the tallest object during the robot's movement in the picking motion.



Hence, an optimal solution should be able to predict objects which are at the topmost position and pick the box with highest height.

General steps of solution:

1. Get objects from computeObjects()
2. Obtain a list of objects which are stacked on top each other.
3. For the stack of objects, calculate the height of the box with respect to surface.
4. For the objects which are at the topmost in their center -> (x,y) and obtain their heights.
5. From the above objects find the object with highest height.

Pseudo-code:

```
Objects = compute_objects()

# Obtain a list of objects which are stacked on top each other
for i in objects
    list_base.append(i)
    list_objects = [i]
    for j in objects['center_x']
       if i not equal to j and objects['center_z'][i] < objects['cen-
ter_z'][j]
          if overlap(objects['center_x'][i], objects['center_y'][i], ob-
jects['x'][i], objects['y'][i],
                           objects['center_x'][j], objects['center_y'][j],
objects['x'][j], objects['y'][j])
                    list_objects.append(j)
                    dict_objects["i"] = list_objects
    dict_base['base'] = list_base
    dict_objects <- dict_objects + dict_base

# For the stack of objects, calculate the height of the box with respect to
surface
# For the objects which are at the topmost in their center -> (x,y)  and
obtain their heights.

for key, value in dict_objects
   if key not 'base'
     repeat
        if h_init = 0
           h_init = 2 * objects['center_z'][value[i]]
        else:
           h_init = h_init + 2 * (objects['center_z'][value[i]] - h_init)
        until length(value)
        alpha["key"] = h_init
    else
      for i in value
         h_in <- 2 * objects['center_z'][value[i]]
                value_list <- value[i]
            v_index =index of h_in (max(h_in))
            alpha["value"] = max(h_in)
       # From the above objects find the object with highest height.
       max_value = max(alpha)
       print('pick object:', max_value)
```