```
text_box_coordinates = [
    {"x1": sublist[0], "y1": sublist[1], "x2": sublist[2], "y2": sublist[3]} for sublist in text_box_coordinates
]
parent_box_coordinates = [
    {"x1": sublist[0], "y1": sublist[1], "x2": sublist[2], "y2": sublist[3]} for sublist in parent_box_coordinates
]

# def check_all_intersection():
#     return True or False

def check_intersection(coords):
    return False

# def get_alignment(child, parent, distance):
#     #             _____ x2, y2
#     #      x1, y1_____
#     # _________________6_____px2, py2
#     #
#     #
#     # _________________6_____px1, py1
#     #       _____ x2, y2
#     # x1, y1_____
#     if child.y1 == (parent.y2 + distance) or parent.y1 - 6 == child.y2:
#         return 'horizontal'
#     return 'vetical'

def get_position(child, parent, distance):
    #         px2, py2
    # _________       x2, y2
    #          |  |  |
    #          |  |  |
    #          |  |  |
    #            x1, y1
    if parent.x2 + distance == child.x1:
        return 'right'
    elif parent.y1 - distance == child.y2:
        return 'bottom'
    elif child.y1 == parent.y2 + distance:
        return 'top'
    else:
        return 'left'

def move_horizontal_left(child, moving_rate):
    child.x1 -= moving_rate
    child.x2 -= moving_rate
    return child

def move_horizontal_right(child, moving_rate):
    child.x1 += moving_rate
    child.x2 += moving_rate
    return child

def move_vertical_up(child, moving_rate):
    child.y1 += moving_rate
    child.y2 += moving_rate
    return child

def move_vertical_down(child, moving_rate):
    child.x1 -= moving_rate
    child.x2 -= moving_rate
    return child

def change_course(child,  distance, position):
    new_child = {'x1': 0, 'y1': 0, 'x2': 0, 'y2': 0}
    x1, y1, x2, y2 = child.values()
    if position == 'top':
        #                        5,  7
        #           ----------- x2, y2     |   | x2, y2
        #    x1, y1 -----------             |   |
        #    2,  3                          |   |
        #  -- change course                 |   |
        #                                 x1, y1
        new_child.x1 =  x1 + (x2 - x1) + distance
        new_child.x2 =  x2 + distance + (y2 - y1)
        new_child.y1 =  y1 - ((y2-y1) - (x2 -x1))
```

```
        new_child.y2 =  y2
    # Warning!!! Need to change the below coordinates
    elif position == 'bottom':
        #                       5,  7
        #           ----------- x2, y2     |   | x2, y2
        #    x1, y1 -----------             |   |
        #    2,  3                          |   |
        #  -- change course                 |   |
        #  if already horizontal           x1, y1
        new_child.x1 =  x1 + (x2 - x1) + distance
        new_child.x2 =  x2 + distance + (y2 - y1)
        new_child.y1 =  y1 - ((y2-y1) - (x2 -x1))
        new_child.y2 =  y2
    elif position == 'left':
        #                       5,  7
        #           ----------- x2, y2     |   | x2, y2
        #    x1, y1 -----------             |   |
        #    2,  3                          |   |
        #  -- change course                 |   |
        #  if already horizontal           x1, y1
        new_child.x1 =  x1 + (x2 - x1) + distance
        new_child.x2 =  x2 + distance + (y2 - y1)
        new_child.y1 =  y1 - ((y2-y1) - (x2 -x1))
        new_child.y2 =  y2
    else:
        new_child.x1 =  x1 + (x2 - x1) + distance
        new_child.x2 =  x2 + distance + (y2 - y1)
        new_child.y1 =  y1 - ((y2-y1) - (x2 -x1))
        new_child.y2 =  y2
    return new_child

def show_board(child, parent):
    pass
# algo
# while not check_all_intersection:
# for i in range(len(text_box_coordinates)):
i = 100
moving_rate = 0.123
distance = 6
while not found_flag:
    child = text_box_coordinates[i]
    parent = parent_box_coordinates[i]
    # alignment = get_alignment(child, parent)
    position = get_position(child, parent, distance)
    if position == 'top':
        # increase x1 and x2 values by 0.1% of (x2-x1)
        child = move_horizontal_right(child, moving_rate)
        if child.x2 >= parent.x2:
            child = change_course(child, distance, position)

    if position == 'bottom':
        child = move_horizontal_left(child, moving_rate)
        if child.x1 <= parent.x1:
            child = change_course(child, distance, position)

    if position == 'left':
        child = move_vertical_up(child, moving_rate)
        if child.y2 >= parent.y2:
            child = change_course(child, distance, position)

    if position == 'right':
        child = move_vertical_down(child, moving_rate)
        if child.y1 <= parent.y1:
            child = change_course(child, distance, position)

    found_flag = check_intersection(child)

    show_board(child, parent)
    i -= 1
    if i < 0:
        break
    # found_all_flag = check_all_intersection()
```