Technical Report: Robotic Manipulation Trajectory Planning and Execution

1. Introduction

This report provides a detailed description of the approach and solution in planning and executing a robotic manipulation task using the KUKA LBR iiwa 7 R800 robotic arm. The project's objective was to design and compute a precise trajectory enabling the robot's end-effector to accurately position a rectangular object onto predefined targets, maintaining consistent orientation throughout the process.

2. Robot and Environment Setup

The robot model was loaded using MATLAB's Robotics System Toolbox from the URDF file:

```
robot = importrobot('iiwa7.urdf');
robot.DataFormat = 'row';
eeName = robot.BodyNames{end};
```

The end-effector orientation was consistently fixed (Z-down and X-forward) using Euler angles:

```
yawAlign = -pi/2;
Rfixed = eul2rotm([yawAlign, 0, pi], 'ZYX');
```

3. Coordinate Transformations

- End-Effector Offset:
  A translation offset from the Tool Center Point (TCP) to the manipulated shape was applied:

```
shapeOffset = [12.5, -40.5, -15]/1000;  % in meters
T_tcp_shape = trvec2tform(shapeOffset);
```

- Hand-Eye Calibration (Tec):
  Defined to represent the camera's pose relative to the TCP:

```
Tec = [ 0  1 0     0;
       -1  0 0 -0.0662;
        0  0 1  0.0431;
        0  0 0     1 ];
```

- Marker Localization (Tca):
  Computed using solvePnP, defining the marker's pose relative to the camera:

```
R_ca = eul2rotm(deg2rad([-37.6880, -4.9345, 178.5475]), 'ZYX');
t_ca = [-0.019795; -0.0331015; 0.6206805];
Tca  = [R_ca, t_ca; 0 0 0 1];
```

- Base-to-Marker Transformation:
  Calculated using the known robot joint configuration at the initial photo-taking position (qc_photo):

```
T_base_tcp_c = getTransform(robot, qc_photo, eeName);
T_base_cam = T_base_tcp_c / Tec;
T_base_marker = T_base_cam / Tca;
```

4. Target and Marker Grid Definition

Marker positions were explicitly defined relative to the robot's base coordinate frame, considering precise geometric offsets including marker dimensions and spacing for accurate placement:

```
markerOffsets = [
  -dx/2, -(mw/2 + hgap + dy/2), 0;
   dx/2, -(mw/2 + hgap + dy/2), 0;
   dx/2, -(mw/2 + hgap - dy/2), 0;
  -dx/2, -(mw/2 + hgap - dy/2), 0
];
```

5. Waypoint Generation

Nine Cartesian waypoints were generated, ensuring smooth transitions:

- Initial Position: TCP position at photo capture
- Intermediate Positions: Alternating safe (raised) and actual target positions to avoid collisions

6. Path Interpolation

A Cartesian path was generated using linear interpolation over 20 seconds with intervals of 5 ms, providing smooth continuous motion:

```
t_wp = linspace(0, t_total, size(pos_list,2));
t_cmd = (0:dt_cmd:t_total)';

x_cmd = interp1(t_wp, pos_list(1,:), t_cmd)';
y_cmd = interp1(t_wp, pos_list(2,:), t_cmd)';
z_cmd = interp1(t_wp, pos_list(3,:), t_cmd)';
```

7. Inverse Kinematics

Inverse kinematics were computed at each waypoint using MATLAB's built-in solver, ensuring smooth and consistent end-effector motion:

```
ik = inverseKinematics('RigidBodyTree', robot);
weights = [0.1 0.1 0.1 1 1 1];
[qSol,~] = ik(eeName, T_targetTCP, weights, q_prev);
```

8. Simulation and Visualization

A simulation visualized the trajectory at an accelerated playback speed to confirm the correctness and feasibility of computed trajectories:

```
simSpeedFactor = 10;
for k = 1:frameStep:length(t_cmd)
    show(robot, q_cmd(:,k)', 'Frames','off','PreservePlot', false, 'Parent', ax);
    drawnow limitrate;
end
```

9. Trajectory Recording

The trajectory was recorded into a text file (singareddy rajeev.txt) with precise formatting suitable for robotic controllers:

```
fprintf(fid, '%.8f %.8f %.8f %.8f %.8f %.8f %.8f\n', q_cmd(:,i));
```

10. Results Analysis

Joint angles and velocities were systematically analysed and plotted, ensuring trajectory smoothness, adherence to joint limits, and feasibility for practical execution.

11. Conclusion

The developed methodology successfully addressed the robotic manipulation task by integrating precise trajectory planning, robust inverse kinematics, and effective collision avoidance strategies, ensuring accurate target placement and safe, efficient robotic operations.