Answer Sheet

Index No: 210503H

1. What is the geometrical structure of the Puma 560 robot?

   6 axis, RRRRRR, stdDH, slowRNE

2. Fill the DH parameter table for Puma 560 robot.

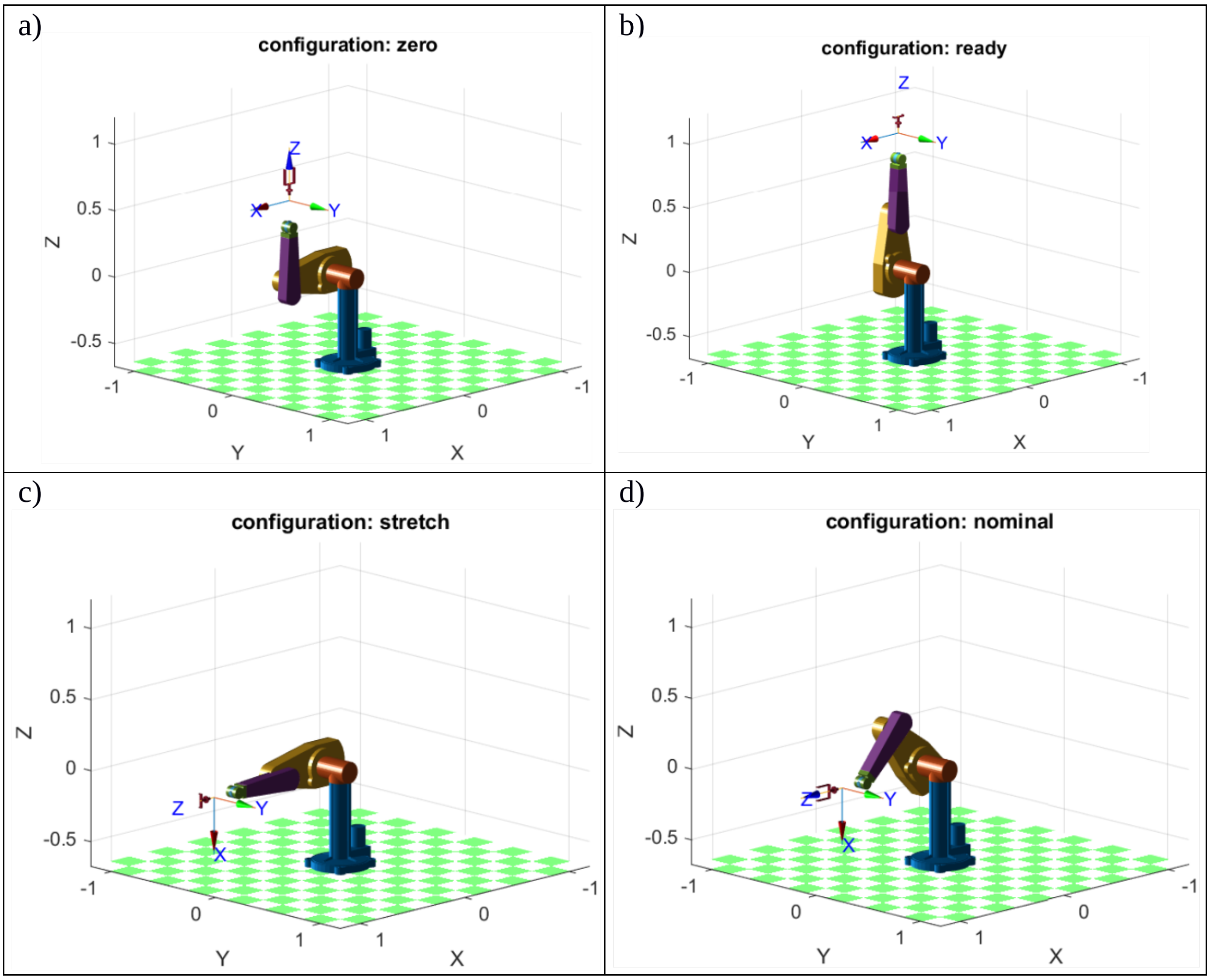
j	θ	d	a	α
1	q1	0	0	1.5708
2	q2	0	0.4318	0
3	q3	0.15005	0.0203	-1.5708
4	q4	0.4318	0	1.5708
5	q5	0	0	-1.5708
6	q6	0	0	0

3. Joint coordinate vectors of Puma 560 robot for the following canonical configurations:
   a) zero angle: qz = [0 0 0 0 0 0]
   b) ready : qr = [0 1.5708 -1.5708 0 0 0]
   c) stretch : qs = [0 0 -1.5708 0 0 0]
   d) nominal : qn = [0 0.7854 3.1416 0 0.7854 0]

4. Forward kinematics for tool center point (TCP) in Procedure 3.5 for the canonical configurations.

Configuration	Position	Orientation (Rotation Matrix)
Zero angle	0.4521 -0.15 0.6318	1 0 0 0 1 0 0 0 1
Ready	0.0203 -0.15 1.064	1 0 0 0 1 0 0 0 1
Stretch	1.064 -0.1501 -0.0203	0 0 1 0 1 0 -1 0 0
Nominal	0.5963 -0.1501 -0.01435	0 0 1 0 1 0 -1 0 0

5. Visualization of Puma 560 robot for Procedure 3.6.



6. Inverse kinematics joint vector for Procedure 3.8. What is your observation?

   [2.6486 -3.9270 0.0940 2.5326 0.9743 0.3734]

   This joint arrangement successfully attains the required end-effector position for the nominalconfiguration.

7. Correct arm configuration for Procedure 3.9.

   The correct arm configuration is the left-hand, elbow-up (lu)configuration.

8. What can be observed for Procedure 3.10?

   When an unreachable point (10 meters away along the x-axis) is provided, the inverse kinematics solver returns NaN (Not a Number) values for all joint angles. This indicates that the robot cannot reach the specified point. Additionally, a warning message is displayed: "Warning: pointnotreachable."

9. MATLAB code for the entire procedure.

```
% 3.1 instance of puma560
mdl_puma560;

% 3.2 dh
p560;

% 3.3 joint coord vec
qz = p560.qz;  % zero angle
qr = p560.qr;  % ready
qs = p560.qs;  % stretch
qn = p560.qn;  % nominal

% 3.4 tool transform
T_tool = SE3(0, 0, 0.2);  % 200mm extension in z-direction
p560.tool = T_tool;

% 3.5 fk
T_zero = p560.fkine(qz);
T_ready = p560.fkine(qr);
T_stretch = p560.fkine(qs);
T_nominal = p560.fkine(qn);

% 3.6 canonical config realistic plots
can_configs = {qz, qr, qs, qn};
can_configs_str = {'zero', 'ready', 'stretch', 'nominal'};
figure('Name', 'Canonical Configs', 'Position', [100, 100, 1200, 900]);
for i = 1:length(can_configs)
    subplot(2, 2, i)
    p560_config = SerialLink(p560, 'name', ['p560_', can_configs_str{i}]);
    p560_config.plot3d(can_configs{i})
    title(['configuration: ', can_configs_str{i}])
end

% inverse kinematics
% 3.7 reset the tool
p560.tool = SE3();  % reset to identity transform

% 3.8 ik for fk or nominal config
T_nominal = p560.fkine(qn);
q_inv = p560.ikine6s(T_nominal);
disp('inverse kinematics result:')
disp(q_inv)

% inverse kinematics experiment
% 3.9 correct arm geometry for nominal
configs = {'lu', 'ld', 'ru', 'rd'};
figure('Name', 'Arm Geometries', 'Position', [100, 100, 1200, 900]);
for i = 1:length(configs)
    q = p560.ikine6s(T_nominal, configs{i});

    % display the joint angles
    disp(['configuration: ', configs{i}])
    disp(q)

    subplot(2, 2, i)
    p560_config = SerialLink(p560, 'name', ['p560_', configs{i}]);
    p560_config.plot3d(q)

    title(['configuration: ', configs{i}])
end

% 3.10 unreachable test
T_unreachable = SE3(10, 0, 0);  % a point 10 meters away on x-axis
q_unreach = p560.ikine6s(T_unreachable);
disp('inverse kinematics for unreachable point:')
disp(q_unreach)
```

10. Explain in point form what the MATLAB code in 3.11 does.

1. Close all open figures and clear workspace.
2. Load the Puma 560 robot model.
3. Define initial end-effector pose at (0.8, 0, 0) with a 90° rotation about the Y-axis.
4. Define final end-effector pose at (-0.8, 0, 0) with a 180° rotation about the X-axis.
5. Solve inverse kinematics for the initial pose to get joint configuration q1.
6. Solve inverse kinematics for the final pose to get joint configuration q2.
7. Create a time vector from 0 to 2 seconds with 0.05-second intervals.
8. Generate a joint-space trajectory between q1 and q2.
9. Plot and animate the 3D robottrajectory.