
```

% UR5 co-robot kinematics analysis
% ME5250 project 2 - topic 1
% Letter K Trajectory

clear; clc; close all;

function print_matrix(M)
    for i = 1:size(M,1)
        fprintf(' [%10.6f %10.6f %10.6f %10.6f]\n', M(i,:));
    end
    fprintf('\n');
end

% DH parameters
fprintf('UR5 DH Parameters\n');
DH.a = [0, -0.425, -0.39225, 0, 0, 0];
DH.d = [0.089159, 0, 0, 0.10915, 0.09465, 0.0823];
DH.alpha = [pi/2, 0, 0, pi/2, -pi/2, 0];
DH.n = 6;

%DH function
function T = dh_matrix(a, d, alpha, theta)
    ct = cos(theta); st = sin(theta);
    ca = cos(alpha); sa = sin(alpha);
    T = [ct, -st*ca, st*sa, a*ct;
          st, ct*ca, -ct*sa, a*st;
          0, sa, ca, d;
          0, 0, 0, 1];
end

function [T, T_all] = fk_ur5(q, DH)
    T = eye(4);
    T_all = zeros(4,4,DH.n+1);
    T_all(:, :, 1) = T;
    for i = 1:DH.n
        Ti = dh_matrix(DH.a(i), DH.d(i), DH.alpha(i), q(i));
        T = T * Ti;
        T_all(:, :, i+1) = T;
    end
end

function r = rot2aa(R)
    theta = acos(max(-1, min(1, (trace(R)-1)/2)));
    if abs(theta) < 1e-10
        r = [0;0;0];
    elseif abs(theta-pi) < 1e-10
        [~,idx] = max(diag(R));
        v = R(:,idx) + [idx==1;idx==2;idx==3];
        v = v/norm(v);
        r = theta*v;
    else

```

```

        r = theta/(2*sin(theta))*[R(3,2)-R(2,3); R(1,3)-R(3,1); R(2,1)-
R(1,2)];
    end
end

fprintf('Joint | a (m) | d (m) | alpha (rad) | alpha (deg)\n');
for i = 1:DH.n
    fprintf(' %d | %10.5f | %10.5f | %11.5f | %10.0f\n', ...
        i, DH.a(i), DH.d(i), DH.alpha(i), rad2deg(DH.alpha(i)));
end
fprintf('\n');

% fk
% jacobian
function J = jacobian_ur5(q, DH)
    [~, T_all] = fk_ur5(q, DH);
    pe = T_all(1:3,4,end);
    J = zeros(6, DH.n);
    for i = 1:DH.n
        Ti = T_all(:, :, i);
        zi = Ti(1:3,3);
        pi = Ti(1:3,4);
        J(1:3,i) = cross(zi, pe - pi);
        J(4:6,i) = zi;
    end
end

fprintf('Analytical fk\n');

q_home = zeros(6,1);
[T_home, T_all] = fk_ur5(q_home, DH);

fprintf('At zero/home position (theta = zero):\n');
fprintf('End-effector T06:\n');
print_matrix(T_home);

q_test = [pi/2; -pi/6; pi/4; -pi/4; pi/6; -pi/2];
[T_test, ~] = fk_ur5(q_test, DH);
fprintf('Test config q = [90, -30, 45, -45, 30, -90] deg:\n');
fprintf('End-effector T matrix:\n');
print_matrix(T_test);

% K trajectory
fprintf('Letter K Trajectory with 0.005m increments\n');

spacing = 0.01; %increment = 10mm or 0.01m
cx = 0.3; cy = 0.25; cz = 0.5; %starting coordinates
R_fixed = [0 0 1; 0 1 0; -1 0 0]; %tool pointing direction = +x axis

% Drawing order:
% start at middle point, go up 100mm, go down 100mm
% go up-right at 45 deg for 100mm, return to middle, go down-right at 45 deg
for 100mm

```

```

leg_length = 0.1; % each segment = 100mm

% starting point of K trajectory
mid_y = cy;
mid_z = cz;

top_point = [cx, mid_y, mid_z + leg_length]; % top of vertical
bottom_point = [cx, mid_y, mid_z - leg_length]; % bottom of vertical
upper_diag = [cx, mid_y + leg_length*cos(pi/4), mid_z + leg_length*sin(pi/4)]; % up-right end (45 deg)
lower_diag = [cx, mid_y + leg_length*cos(pi/4), mid_z - leg_length*sin(pi/4)]; % down-right end (-45 deg)
middle_point = [cx, mid_y, mid_z];

% Number of points per segment
n_seg = round(leg_length / spacing);

traj = [];

% Segment 1: midpoint to up
for i = 0:n_seg
    t = i / n_seg;
    pt = middle_point + t * (top_point - middle_point);
    traj = [traj; pt];
end

% Segment 2: up to midpoint
for i = 1:n_seg
    t = i / n_seg;
    pt = top_point + t * (middle_point - top_point);
    traj = [traj; pt];
end

% Segment 3: midpoint to down
for i = 1:n_seg
    t = i / n_seg;
    pt = middle_point + t * (bottom_point - middle_point);
    traj = [traj; pt];
end

% Segment 4: down to midpoint
for i = 1:n_seg
    t = i / n_seg;
    pt = bottom_point + t * (middle_point - bottom_point);
    traj = [traj; pt];
end

% Segment 5: midpoint to up-right
for i = 1:n_seg
    t = i / n_seg;
    pt = middle_point + t * (upper_diag - middle_point);
    traj = [traj; pt];
end

```

```

% Segment 6: up-right to midpoint
for i = 1:n_seg
    t = i / n_seg;
    pt = upper_diag + t * (middle_point - upper_diag);
    traj = [traj; pt];
end

% Segment 7: midpoint to down-right
for i = 1:n_seg
    t = i / n_seg;
    pt = middle_point + t * (lower_diag - middle_point);
    traj = [traj; pt];
end

n_traj = size(traj,1);

fprintf('K Letter Lengths:\n');
fprintf('Verticle length: %.0f m\n', leg_length);
fprintf('Diagonal angles: 45 and -45 degrees\n');
fprintf('Initial point (K): [%.3f, %.3f, %.3f] m\n', middle_point);
fprintf('Plane: YZ (X constant at %.3f m)\n', cx);
fprintf('Total points: %d\n\n', n_traj);

fprintf('End points:\n');
fprintf('  Midpoint: [%.4f, %.4f, %.4f]\n', middle_point);
fprintf('  Up:      [%.4f, %.4f, %.4f]\n', top_point);
fprintf('  Down:    [%.4f, %.4f, %.4f]\n', bottom_point);
fprintf('  Up-tight end: [%.4f, %.4f, %.4f]\n', upper_diag);
fprintf('  Down-right end: [%.4f, %.4f, %.4f]\n\n', lower_diag);

fprintf('First 30 points:\n');
for i = 1:min(30,n_traj)
    fprintf('  %3d: [%.4f, %.4f, %.4f]\n', i, traj(i,:));
end
fprintf('\n');

% ik solver
fprintf('Newton''s IK Solver\n');

%ik funtion
function [q_sol, success, iters] = newton_ik(T_tgt, q0, DH)
    max_it = 100; tol = 1e-5; lam = 0.05;
    q = q0(:); success = false;
    for iters = 1:max_it
        [Tc,~] = fk_ur5(q, DH);
        e = pose_err(Tc, T_tgt);
        if norm(e(1:3)) < tol && norm(e(4:6)) < tol
            success = true; break;
        end
        J = jacobian_ur5(q, DH);
        Jd = J' / (J*J' + lam^2*eye(6));
        q = q + Jd * e;
    end
    q_sol = mod(q+pi, 2*pi) - pi;

```

```

end

%end-effector position error
function e = pose_err(T_curr, T_tgt)
    ep = T_tgt(1:3,4) - T_curr(1:3,4);
    Re = T_tgt(1:3,1:3) * T_curr(1:3,1:3)';
    er = rot2aa(Re);
    e = [ep; er];
end

%initial tracking point
q_init = [0; -pi/4; pi/2; -pi/4; pi/2; 0];
q_solutions = zeros(n_traj, 6);
success_flags = zeros(n_traj, 1);
iter_counts = zeros(n_traj, 1);
q_curr = q_init;

fprintf('Solving IK for %d points...\n', n_traj);
tic;
for i = 1:n_traj
    T_target = eye(4);
    T_target(1:3,1:3) = R_fixed;
    T_target(1:3,4) = traj(i,:)';

    [q_sol, ok, iters] = newton_ik(T_target, q_curr, DH);
    q_solutions(i,:) = q_sol';
    success_flags(i) = ok;
    iter_counts(i) = iters;
    if ok, q_curr = q_sol; end

    if mod(i,20)==0
        fprintf(' %d/%d done (%.0f%% success)\n', i, n_traj,
100*mean(success_flags(1:i)));
    end
end
t_solve = toc;

fprintf('\nResults:\n');
fprintf(' Success: %.1f%% (%d/%d)\n', 100*mean(success_flags),
sum(success_flags), n_traj);
fprintf(' Avg iterations: %.1f\n', mean(iter_counts));
fprintf(' Total time: %.3f s (%.4f s/point)\n\n', t_solve, t_solve/n_traj);

% Print IK solutions for first 30 points
fprintf('IK Solutions (first 30 points) - Joint angles in degrees:\n');
fprintf(' Pt | q1 | q2 | q3 | q4 | q5 | q6 | \n');
for i = 1:min(30, n_traj)
    q_deg = rad2deg(q_solutions(i,:));
    fprintf(' %2d | %8.2f | %8.2f | %8.2f | %8.2f | %8.2f | %8.2f | \n', ...
i, q_deg(1), q_deg(2), q_deg(3), q_deg(4), q_deg(5), q_deg(6));
end
fprintf('\n');

```

```

% 3D stick figure
fprintf('3D Stick Figure Animation\n');
fig1 = figure('Name','UR5 Animation - Letter K','Position',[100 100 1200
800]);
ee_path = zeros(n_traj,3);
skip = 2;

fprintf('Animating (every %d frames)... \n', skip);
for fr = 1:skip:n_traj
    if ~isvalid(fig1), break; end
    clf(fig1);

    q = q_solutions(fr,:);
    [T_ee, T_all] = fk_ur5(q, DH);
    ee_path(fr,:) = T_ee(1:3,4);

    subplot(1,2,1); hold on; grid on; axis equal;

    % Joint positions
    jp = zeros(7,3);
    for k = 1:6, jp(k+1,:) = T_all(1:3,4,k+1); end

    % Links and joints
    plot3(jp(:,1),jp(:,2),jp(:,3),'b-','LineWidth',3);
    plot3(jp(:,1),jp(:,2),jp(:,3),'ko','MarkerSize',10,'MarkerFaceColor','k');

    % End-Effector frame (RGB=XYZ)
    p = T_ee(1:3,4); s = 0.05;

    quiver3(p(1),p(2),p(3),T_ee(1,1)*s,T_ee(2,1)*s,T_ee(3,1)*s,'r','LineWidth',2);
    quiver3(p(1),p(2),p(3),T_ee(1,2)*s,T_ee(2,2)*s,T_ee(3,2)*s,'g','LineWidth',2);
    quiver3(p(1),p(2),p(3),T_ee(1,3)*s,T_ee(2,3)*s,T_ee(3,3)*s,'b','LineWidth',2);

    % Trajectory and path
    plot3(traj(:,1),traj(:,2),traj(:,3),'m--','LineWidth',1);
    vp = ee_path(1:skip:fr,:); vp = vp(any(vp,2),:);
    if size(vp,1)>1, plot3(vp(:,1),vp(:,2),vp(:,3),'c-','LineWidth',2); end

    % Base frame
    quiver3(0,0,0,0.1,0,0,'r','LineWidth',1.5);
    quiver3(0,0,0,0,0.1,0,'g','LineWidth',1.5);
    quiver3(0,0,0,0,0,0.1,'b','LineWidth',1.5);

    xlabel('X (m)'); ylabel('Y (m)'); zlabel('Z (m)');
    title(sprintf('UR5 - Letter K - Frame %d/%d',fr,n_traj));
    xlim([-0.2 0.8]); ylim([-0.5 0.5]); zlim([-0.2 0.8]);
    view(45,30);

    subplot(1,2,2);
    bar(rad2deg(q)); xlabel('Joint'); ylabel('Angle (deg)');
    title('Joint Angles'); ylim([-180 180]); grid on;

```

```

drawnow; pause(0.02);
end
fprintf('Animation complete.\n\n');

UR5 DH Parameters
Joint / a (m) / d (m) / alpha (rad) / alpha (deg)
1 / 0.00000 / 0.08916 / 1.57080 / 90
2 / -0.42500 / 0.00000 / 0.00000 / 0
3 / -0.39225 / 0.00000 / 0.00000 / 0
4 / 0.00000 / 0.10915 / 1.57080 / 90
5 / 0.00000 / 0.09465 / -1.57080 / -90
6 / 0.00000 / 0.08230 / 0.00000 / 0

```

Analytical fk

At zero/home position (theta = zero):

End-effector T06:

```

[ 1.000000  0.000000  0.000000 -0.817250]
[ 0.000000  0.000000 -1.000000 -0.191450]
[ 0.000000  1.000000  0.000000 -0.005491]
[ 0.000000  0.000000  0.000000  1.000000]

```

Test config $q = [90, -30, 45, -45, 30, -90]$ deg:

End-effector T matrix:

```

[ 0.000000  0.500000  0.866025  0.180424]
[ -0.500000  0.750000 -0.433013 -0.829907]
[ -0.866025 -0.433013  0.250000  0.138743]
[ 0.000000  0.000000  0.000000  1.000000]

```

Letter K Trajectory with 0.005m increments

K Letter Lengths:

Verticle length: 0 m

Diagonal angles: 45 and -45 degrees

Initial point (K): [0.300, 0.250, 0.500] m

Plane: YZ (X constant at 0.300 m)

Total points: 71

End points:

```

Midpoint: [0.3000, 0.2500, 0.5000]
Up:       [0.3000, 0.2500, 0.6000]
Down:     [0.3000, 0.2500, 0.4000]
Up-tight end: [0.3000, 0.3207, 0.5707]
Down-right end: [0.3000, 0.3207, 0.4293]

```

First 30 points:

```

1: [0.3000, 0.2500, 0.5000]
2: [0.3000, 0.2500, 0.5100]
3: [0.3000, 0.2500, 0.5200]
4: [0.3000, 0.2500, 0.5300]
5: [0.3000, 0.2500, 0.5400]
6: [0.3000, 0.2500, 0.5500]
7: [0.3000, 0.2500, 0.5600]
8: [0.3000, 0.2500, 0.5700]
9: [0.3000, 0.2500, 0.5800]

```

```

10: [0.3000, 0.2500, 0.5900]
11: [0.3000, 0.2500, 0.6000]
12: [0.3000, 0.2500, 0.5900]
13: [0.3000, 0.2500, 0.5800]
14: [0.3000, 0.2500, 0.5700]
15: [0.3000, 0.2500, 0.5600]
16: [0.3000, 0.2500, 0.5500]
17: [0.3000, 0.2500, 0.5400]
18: [0.3000, 0.2500, 0.5300]
19: [0.3000, 0.2500, 0.5200]
20: [0.3000, 0.2500, 0.5100]
21: [0.3000, 0.2500, 0.5000]
22: [0.3000, 0.2500, 0.4900]
23: [0.3000, 0.2500, 0.4800]
24: [0.3000, 0.2500, 0.4700]
25: [0.3000, 0.2500, 0.4600]
26: [0.3000, 0.2500, 0.4500]
27: [0.3000, 0.2500, 0.4400]
28: [0.3000, 0.2500, 0.4300]
29: [0.3000, 0.2500, 0.4200]
30: [0.3000, 0.2500, 0.4100]

```

Newton's IK Solver

Solving IK for 71 points...

20/71 done (95% success)

40/71 done (98% success)

60/71 done (98% success)

Results:

Success: 98.6% (70/71)

Avg iterations: 5.7

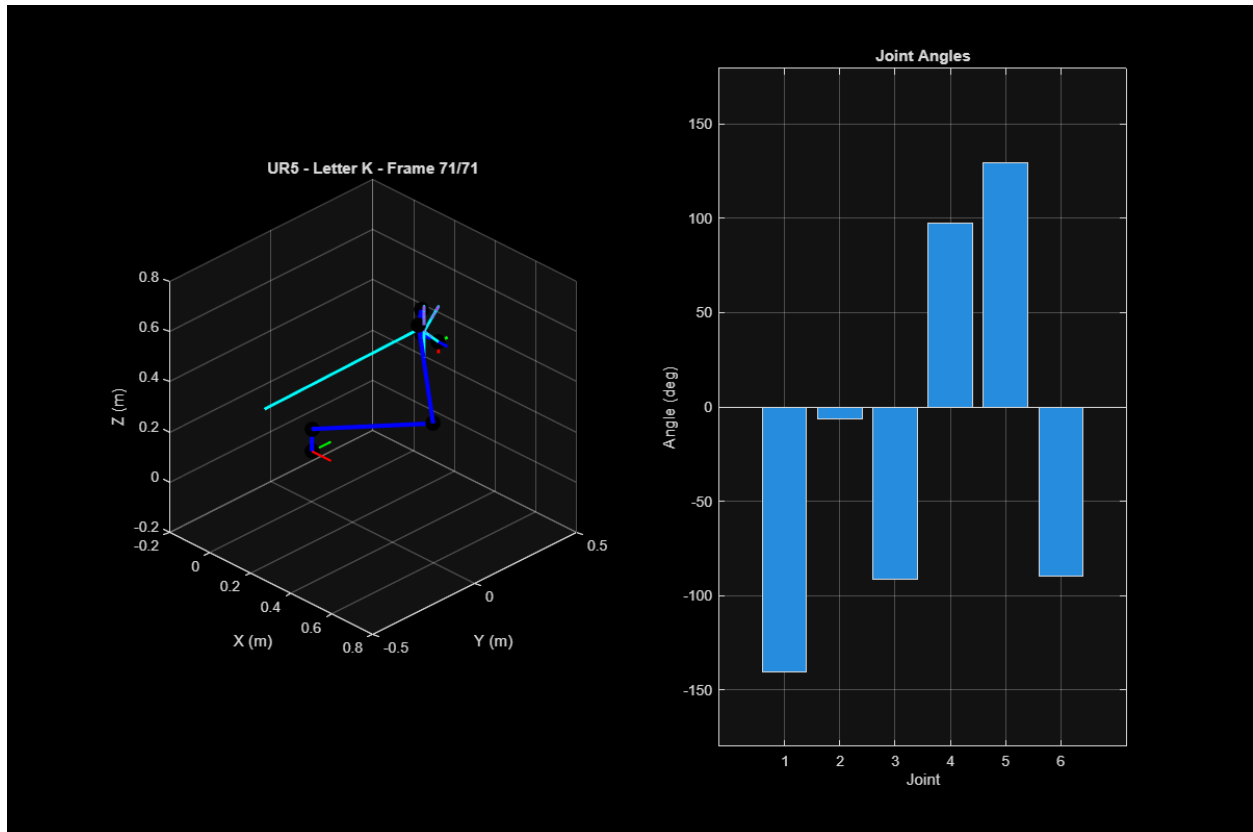
Total time: 0.021 s (0.0003 s/point)

IK Solutions (first 30 points) - Joint angles in degrees:

Pt	q1	q2	q3	q4	q5	q6
1	-150.66	122.90	113.52	122.96	119.30	-90.29
2	-150.27	-18.35	-84.97	103.32	119.73	-90.00
3	-150.27	-19.67	-83.17	102.84	119.73	-90.00
4	-150.27	-21.00	-81.33	102.33	119.73	-90.00
5	-150.27	-22.33	-79.45	101.78	119.73	-90.00
6	-150.27	-23.68	-77.52	101.20	119.73	-90.00
7	-150.27	-25.04	-75.54	100.58	119.73	-90.00
8	-150.27	-26.42	-73.50	99.92	119.73	-90.00
9	-150.27	-27.81	-71.41	99.22	119.73	-90.00
10	-150.27	-29.23	-69.25	98.48	119.73	-90.00
11	-150.27	-30.67	-67.03	97.70	119.73	-90.00
12	-150.27	-29.23	-69.25	98.48	119.73	-90.00
13	-150.27	-27.82	-71.41	99.22	119.73	-90.00
14	-150.27	-26.42	-73.50	99.92	119.73	-90.00
15	-150.27	-25.04	-75.54	100.58	119.73	-90.00
16	-150.27	-23.68	-77.52	101.20	119.73	-90.00
17	-150.27	-22.33	-79.45	101.78	119.73	-90.00
18	-150.27	-21.00	-81.33	102.33	119.73	-90.00
19	-150.27	-19.67	-83.17	102.84	119.73	-90.00

20	/	-150.27	/	-18.35	/	-84.97	/	103.32	/	119.73	/	-90.00	/
21	/	-150.27	/	-17.04	/	-86.73	/	103.77	/	119.73	/	-90.00	/
22	/	-150.27	/	-15.73	/	-88.45	/	104.18	/	119.73	/	-90.00	/
23	/	-150.27	/	-14.42	/	-90.14	/	104.56	/	119.73	/	-90.00	/
24	/	-150.27	/	-13.12	/	-91.79	/	104.91	/	119.73	/	-90.00	/
25	/	-150.27	/	-11.81	/	-93.41	/	105.22	/	119.73	/	-90.00	/
26	/	-150.27	/	-10.51	/	-95.00	/	105.50	/	119.73	/	-90.00	/
27	/	-150.27	/	-9.20	/	-96.55	/	105.76	/	119.73	/	-90.00	/
28	/	-150.27	/	-7.90	/	-98.08	/	105.97	/	119.73	/	-90.00	/
29	/	-150.27	/	-6.59	/	-99.57	/	106.16	/	119.73	/	-90.00	/
30	/	-150.27	/	-5.27	/	-101.04	/	106.31	/	119.73	/	-90.00	/

3D Stick Figure Animation
 Animating (every 2 frames)...\br/>
 Animation complete.



Published with MATLAB® R2025b