```
clc;
clear;
%length param
l1 = 0.033;
l2 = 0.1012;
l3 = 0.155;
l4 = 0.1348;
l5 = 0.1087;%0.0974;
l6 = 0.085;

% Angles of
q1 = deg2rad(0);
q2 = deg2rad(0);
q3 = deg2rad(0);
q4 = deg2rad(0);
q5 = deg2rad(0);

n = deg2rad(90);
T1 = [cos(n+q1) -sin(n+q1)*cos(n) sin(n+q1)*sin(n) l1*cos(n+q1); sin(n+q1) cos(n+q1)*cos(n) -cos(n+q1)*sin(n) l1*sin(n+q1); 0 sin(n) cos(n) l2; 0 0 0 1];
n1 = 0;
T2 = [cos(q2) -sin(q2)*cos(n1) sin(q2)*sin(n1) l3*cos(q2); sin(q2) cos(q2)*cos(n1) -cos(q2)*sin(n1) l3*sin(q2); 0 sin(n1) cos(n1) 0; 0 0 0 1];
T3 = [cos(q3) -sin(q3)*cos(n1) sin(q3)*sin(n1) l4*cos(q3); sin(q3) cos(q3)*cos(n1) -cos(q3)*sin(n1) l4*sin(q3); 0 sin(n1) cos(n1) 0; 0 0 0 1];
n = deg2rad(-90);
T4 = [cos(q4) -sin(q4)*cos(n) sin(q4)*sin(n) 0*cos(q4); sin(q4) cos(q4)*cos(n) -cos(q4)*sin(n) 0*sin(q4); 0 sin(n) cos(n) 0; 0 0 0 1];
n =0;
T5 = [cos(q5) -sin(q5)*cos(n) sin(q5)*sin(n) 0*cos(q5); sin(q5) cos(q5)*cos(n) -cos(q5)*sin(n) 0*sin(q5); 0 sin(n) cos(n) l5+l6; 0 0 0 1];
T6 = [0 0 1 0; 0 1 0 0; -1 0 0 0; 0 0 0 1];
Tt = T1*T2*T3*T4*T5;
Of = Tt(1:3,4);
Rf = Tt(1:3,1:3);


vrep=remApi('remoteApi'); % using the prototype file (remoteApiProto.m)
vrep.simxFinish(-1); % just in case, close all opened connections
clientID=vrep.simxStart('127.0.0.1',19999,true,true,5000,5);
if (clientID > -1 )
   disp("Connected");
   [returnCode,joint_0]=vrep.simxGetObjectHandle(clientID,'youBotArmJoint0',vrep.simx_opmode_blocking);
   [returnCode,joint_1]=vrep.simxGetObjectHandle(clientID,'youBotArmJoint1',vrep.simx_opmode_blocking);
   [returnCode,joint_2]=vrep.simxGetObjectHandle(clientID,'youBotArmJoint2',vrep.simx_opmode_blocking);
   [returnCode,joint_3]=vrep.simxGetObjectHandle(clientID,'youBotArmJoint3',vrep.simx_opmode_blocking);
   [returnCode,joint_4]=vrep.simxGetObjectHandle(clientID,'youBotArmJoint4',vrep.
```

```
simx_opmode_blocking);
   [returnCode,gripper1]=vrep.simxGetObjectHandle(clientID,'youBotGripperJoint1',vrep.↙
simx_opmode_blocking);
   [returnCode,gripper2]=vrep.simxGetObjectHandle(clientID,'youBotGripperJoint2',vrep.↙
simx_opmode_blocking);
   i =0;
   while (1)
       [returnCode]=vrep.simxSetJointPosition(clientID,joint_0,q1,vrep.↙
simx_opmode_oneshot); %
       [returnCode]=vrep.simxSetJointPosition(clientID,joint_1,q2-degtorad(90),vrep.↙
simx_opmode_oneshot); %
       [returnCode]=vrep.simxSetJointPosition(clientID,joint_2,q3,vrep.↙
simx_opmode_oneshot); %
       [returnCode]=vrep.simxSetJointPosition(clientID,joint_3,q4+degtorad(90),vrep.↙
simx_opmode_oneshot); %
       [returnCode]=vrep.simxSetJointPosition(clientID,joint_4,q5+degtorad(90),vrep.↙
simx_opmode_oneshot); %
       i = i+1;
       if(i == 200)
           disp("Translation part of Transformation matrix = "); disp(Of);
           [returnCode,position]=vrep.simxGetObjectPosition(clientID,gripper1,joint_0,↙
vrep.simx_opmode_blocking);
           disp("Position of End-effector Obtained from VREP = ");disp(position);
           eul = rotm2eul(Rf);
           disp("Euler Angles Calculated from Rotation Matrix part of Transformation↙
Matrix ");
           disp(Rf);
           [returnCode,eulerAngles]=vrep.simxGetObjectOrientation(clientID,gripper1,↙
joint_0,vrep.simx_opmode_blocking);
           disp("Euler Angles of End-effector Obtained from VREP = ");
           eulerAngles= [eulerAngles(2) eulerAngles(3) eulerAngles(1)];
           rotm = eul2rotm(eulerAngles);
           disp(rotm);
       end

       if(i == 1000)
           %Inverse Kinematics
            Tf =[-0.0868  -0.8660  0.4924  -0.0219; 0.1504  -0.5000  -0.8529 0.0380;↙
0.9848  -0.0000  0.1736  0.3291;  0 0 0 1.0000];
            Of = Tf(1:3,4);
            Rf = Tf(1:3,1:3);
            Rin = Rf;
            ox = Of(1,1);
            oy = Of(2,1);
            oz = Of(3,1);

            xc = ox - (l5+l6)*Rin(1,3);
            yc = oy - (l5+l6)*Rin(2,3);
            zc = oz - (l5+l6)*Rin(3,3);
```

```
            r = sqrt(power(xc,2)+power(yc,2)) - l1;
            %disp(r);
            s = zc - l2;
            %disp(s);
            D = (power(s,2) + power(r,2) - power(l3,2) - power(l4,2))/(2*l3*l4);
            %disp(D);
            a1 =  atan2(-xc,yc);
            if(power(D,2) > 1)
              a3 = 0;
              a2 = 0;
            else
              a3 = atan2(sqrt(1-power(D,2)),D);
              a2 = atan2(s,r) - atan2((l4*sin(a3)),(l3+l4*cos(a3))) ;
            end

            n = deg2rad(90);
            A1 = [cos(n+a1) -sin(n+a1)*cos(n) sin(n+a1)*sin(n) l1*cos(n+a1); sin(n+a1)↙
cos(n+a1)*cos(n) -cos(n+a1)*sin(n) l1*sin(n+a1); 0 sin(n) cos(n) l2; 0 0 0 1];
            n1 = 0;
            A2 = [cos(a2) -sin(a2)*cos(n1) sin(a2)*sin(n1) l3*cos(a2); sin(a2) cos(a2)↙
*cos(n1) -cos(a2)*sin(n1) l3*sin(a2); 0 sin(n1) cos(n1) 0; 0 0 0 1];
            A3 = [cos(a3) -sin(a3)*cos(n1) sin(a3)*sin(n1) l4*cos(a3); sin(a3) cos(a3)↙
*cos(n1) -cos(a3)*sin(n1) l4*sin(a3); 0 sin(n1) cos(n1) 0; 0 0 0 1];

            T03 = A1*A2*A3;
            R03 = T03(1:3,1:3);
            R03t = transpose(R03);
            R35 = R03t*Rin;
            a5 = atan2(R35(3,1),R35(3,2)) + degtorad(180);
            a4 =  atan2(-R35(1,3),R35(2,3));
            if (radtodeg(a5) >= 358)
                a5 =0;
            end
            disp("Angles after Inverse Kinematics for the given Transformation Matrix↙
matrix");
            disp("q1 = ");disp(radtodeg(a1));
            disp("q2 = ");disp(radtodeg(a2));
            disp("q3 = ");disp(radtodeg(a3));
            disp("q4 = ");disp(radtodeg(a4));
            disp("q5 = ");disp(radtodeg(a5));
            q1 = a1;
            q2 = a2;
            q3 = a3;
            q4 = a4;
            q5 = a5;
            i = i+1;
        end

        if(i == 1005)
            disp("Translation part of Transformation matrix given to Inverse Kinematics =↙
"); disp(Of);
```

```
            [returnCode,position]=vrep.simxGetObjectPosition(clientID,gripper1,joint_0,↙
vrep.simx_opmode_blocking);
            disp("Position of End-effector Obtained from VREP = ");disp(position);
            eul = rotm2eul(Rf);
            i = i+1;
        end


    end
   vrep.simxFinish(-1);

end
vrep.delete();
```