The Investigation on the Kinematics, Statics and Dynamics of the ABB IRB-460 Robot

Group 7

March 21, 2021

1 Introduction

In this mini-project, we studied the kinematics, statics and dynamics of ABB IRB-460 robot under the condition of given rotation Angle and angular velocity of three motors by means of MATLAB program calculation and ADAMS simulation software, and compared the data obtained by the two ways. The similarity of the data proved that the result is correct.

2 Position transformation of robot arms

In order to realize the position transformation of the manipulator in space, the rotation of a joint is firstly discussed.

$$R\left(q_{i0},\theta_i\right)=e^{\hat{\xi}_i\theta_i}=\begin{bmatrix} e^{\hat{z}\theta_i} & \left(I-e^{\hat{z}\theta_i}\right)q_{i0} \\ 0 & 1 \end{bmatrix} \tag{1}$$

Where,

$$\hat{\xi}_i=\begin{bmatrix} \hat{\omega}_{i0} & q_{i0}\times\omega_{i0} \\ 0 & 0 \end{bmatrix} \tag{2}$$

q_{i0} is the positive direction of the axis of rotation, and ω_{i0} is the angular velocity.

$$g_{ab}=(\prod_{i=1}^{n}e^{\hat{\xi}_i\theta_i})g_{ab_0} \tag{3}$$

In simplified ABB IRB-460 robot (including base, lower arm, upper arm and toolgrip), n=4.

The following is specific matlab code of this part and functions of ξ_i and $e^{\hat{\xi}_i\theta_i}$.

```
%% Q1--------------------------------------------------------------------------
qMatrix = [[0; 0; 0],[0; 0.260; 0.7425],[0; 0.260; 1.6875],[0; 1.285; 1.6875]];
eps1 = eps([0;0;1],qMatrix(:,1));
eps2 = eps([1;0;0],qMatrix(:,2));
eps3 = eps([1;0;0],qMatrix(:,3));
eps4 = eps([1;0;0],qMatrix(:,4));
exp1 = exp([0;0;1],theta1,qMatrix(:,1));
exp2 = exp([1;0;0],theta2,qMatrix(:,2));
exp3 = exp([1;0;0],(theta3-theta2),qMatrix(:,3));
exp4 = exp([1;0;0],-theta3,qMatrix(:,4));
g_ab0 = [eye(3),[0;1.505;1.470];0,0,0,1];
g_ab = exp1 * exp2 * exp3 * exp4 * g_ab0;
```

Figure 1: The main code of Question 1

```
% eps
function e = eps(w,p)
    p_hat = [0 -p(3) p(2);
             p(3) 0 -p(1);
             -p(2) p(1) 0];
    e = [(p_hat*w);w];
end

% exp = e^(eps_hat*theta)
function ee = exp(axis,theta,q)
    hat = [0 -axis(3) axis(2);
           axis(3) 0 -axis(1);
           -axis(2) axis(1) 0];
    R = eye(3)+hat*sind(theta)+hat^2*(1-cosd(theta));
    ee = [R,(eye(3)-R)*q;0,0,0,1];
end
```

Figure 2: The function code of ξ_i and $e^{\hat{\xi}_i\theta_i}$

3 The forward Jacobian map and the spatial velocity of robot arms

$$\hat{V}_{ab} = \dot{g}_{ab}g_{ab}^{-1} = \hat{\xi}_1\dot{\theta}_1 + e^{\xi_1\theta_1}\hat{\xi}_2 e^{-\xi_1\theta_1}\dot{\theta}_2 + \left(e^{\hat{\xi}_1\theta_1}e^{\hat{\xi}_2\theta_2}\right)\hat{\xi}_3\left(e^{-\hat{\xi}_2\theta_2}e^{-\hat{\xi}_1\theta_1}\right)\dot{\theta}_3 \tag{4}$$

$$V_{ab} = \xi_1\dot{\theta}_1 + \underbrace{Ad_{e^{\hat{\xi}_1\theta_1}}\xi_2}_{\xi_2'}\dot{\theta}_2 + \underbrace{Ad_{e^{\hat{\xi}_1\theta_1}e^{\hat{\xi}_2\theta_2}}\xi_3}_{\xi_3'}\dot{\theta}_3 \tag{5}$$

$$V_{ab} = \underbrace{\begin{bmatrix} \xi_1 & \xi_2' & \xi_3' \end{bmatrix}}_{J}\underbrace{\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}}_{\dot{\theta}} \tag{6}$$

There are two ways to solve $\hat{V}_{ab}$, One is to solve directly by Equation(4), the other is to solve V_{ab} first by Equation(5) and Equation(6), then calculate $\hat{V}_{ab}$. In the calculation process, the second method is adopted.

The following is specific matlab code and function of *hat* calculation of this part.

```
%% Q2-------------------------------------------------------
eps2_t = Ad(exp1)*eps2;
eps3_t = Ad(exp1*exp2)*eps3;
eps4_t = Ad(exp1*exp2*exp3)*eps4;
J = [eps1,(eps2_t-eps3_t),(eps3_t-eps4_t)];
V = J * [theta1_dot; theta2_dot; theta3_dot];
V_hat = E_H(V);
position = [p(g_ab);1];
v_ab = V_hat * position;
```

Figure 3: The main code of Question 2

```
% E_H = eps_hat
function eps_hat = E_H(eps)
    w = [eps(4);eps(5);eps(6)];
    w_hat = [0 -w(3) w(2);
             w(3) 0 -w(1);
             -w(2) w(1) 0];
    v = [eps(1);eps(2);eps(3)];
    eps_hat = [w_hat,v;0,0,0,0];
end
```

Figure 4: The function of *hat* calculation

4 The three joint torques of the robot at a static configuration

Since the robot is under a static configuration, the kinetic energy of the system is 0, that is to say, in the equation $L = T - V$, $T = 0$.

First, we calculate the transformed position of center of mass of each linkage using result of Task 1:

$$c_i = (\prod_{i=1}^{n} e^{\hat{\xi}_i \theta_i}) c_{i0} \tag{7}$$

where the i's are ordered depending on the structure of the linkage connection.

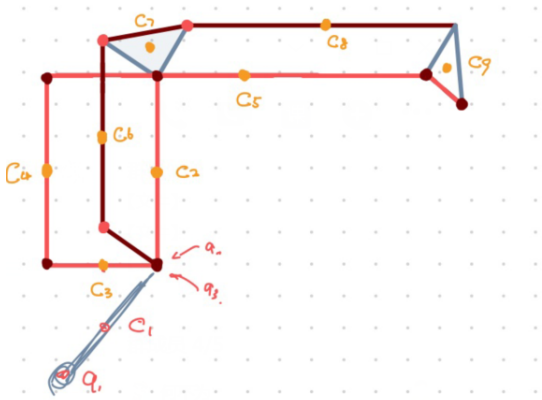

Figure 5: Numbering sequence of mechanical arm links

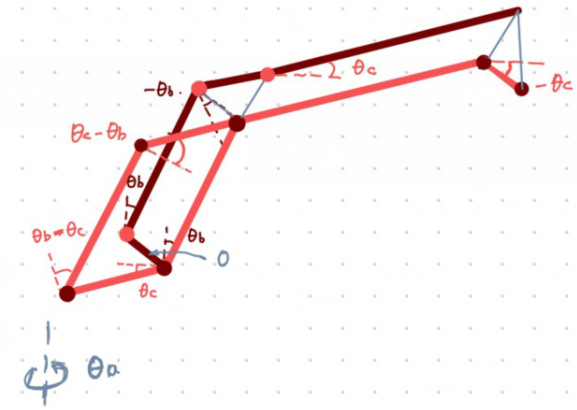

Figure 6: Relationship between drive rotation angle and the rotation angles of each connecting rod

Second, after we get the centroid coordinates of each link, we need to get the partial derivative of L with respect to θ. We tried two ways in this step: one is to take the derivative manually, that is, extract the coefficient in $\prod_{i=1}^{n} e^{\hat{\xi}_i \theta_i}$ to get the partial derivative of L with respect to θ_a, θ_b and θ_c, which are rotation angles of three motors; and then add up these partial derivatives to get the result. The other way is to use a program to compute symbols to get this result.

The torque calculation formula of the manipulator in static state is as follows:

$$\tau = -\frac{\partial L}{\partial \vec{\theta}} + J^T F \tag{8}$$

5 The three joint torques of the robot at a dynamic configuration

In the Task 4, we need to find the torque acting on the actuators when the manipulator is dynamic. The general form of the Euler-Lagrange equation needed to solve this problem is as follows:

$$\tau = \frac{d}{dt}\frac{\partial L}{\partial \dot{\vec{\theta}}} - \frac{\partial L}{\partial \vec{\theta}} + J^T F \tag{9}$$

We use the symbolic calculation used in the third question to deal with the first term of the equation, and then add to the formula for the third question to get the final result.

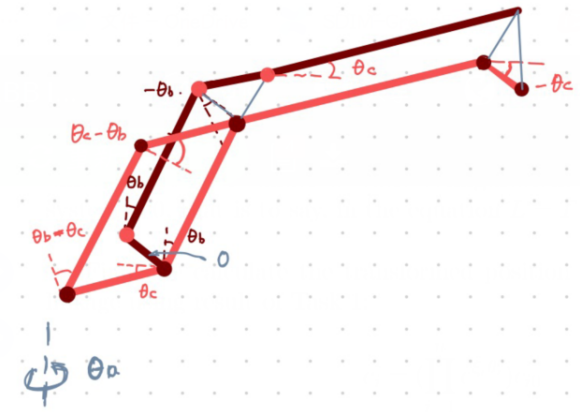

Figure 7: Matlab code for Task 3 and Task 4

6 Simulation verification

In the computer simulation verification, we use MSC Adams multi-body dynamics simulation software to verify our calculation results.

```
5      % angle:
6 –    theta1 = -30;
7 –    theta2 = -80;
8 –    theta3 = -20;
9
10     % angular speed:
11 –   theta1_dot = -30/180*pi;
12 –   theta2_dot = -80/180*pi;
13 –   theta3_dot = -20/180*pi;
14
15     %% Q1----------------------------------------------------------------------------------------
16 –   qMatrix = [[0; 0; 0],[0; 0.260; 0.7425],[0; 0.260; 1.6875],[0; 1.285; 1.6875]];
17 –   eps1 = eps([0;0;1],qMatrix(:,1));
18 –   eps2 = eps([1;0;0],qMatrix(:,2));
19 –   eps3 = eps([1;0;0],qMatrix(:,3));
20 –   eps4 = eps([1;0;0],qMatrix(:,4));
21 –   exp1 = exp([0;0;1],theta1,qMatrix(:,1));
22 –   exp2 = exp([1;0;0],theta2,qMatrix(:,2));
23 –   exp3 = exp([1;0;0],(theta3-theta2),qMatrix(:,3));
24 –   exp4 = exp([1;0;0],-theta3,qMatrix(:,4));
25 –   g_ab0 = [eye(3),[0;1.505;1.470];0,0,0,1];
26 –   g_ab = exp1 * exp2 * exp3 * exp4 * g_ab0
27
28     %% Q2----------------------------------------------------------------------------------------
```

命令行窗口

```
>> Matrix_Arm

g_ab =

    0.8660    0.5000    0.0000    1.1869
   -0.5000    0.8660    0.0000    2.0558
         0   -0.0000    1.0000    0.3385
         0         0         0    1.0000
```

fx

Figure 8: Calculation result for Task 1

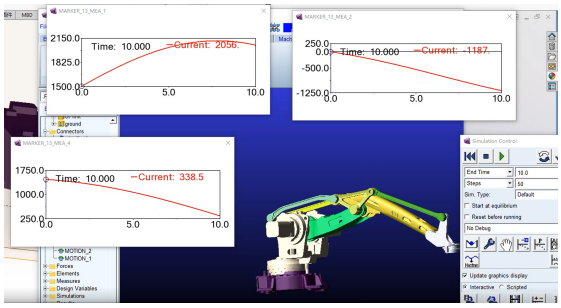

Figure 9: Verification for Task 1

```
14
15      %% Q1----------------------------------------------------------------------
16 -    qMatrix = [[0; 0; 0],[0; 0.260; 0.7425],[0; 0.260; 1.6875],[0; 1.285; 1.6875]];
17 -    eps1 = eps([0;0;1],qMatrix(:,1));
18 -    eps2 = eps([1;0;0],qMatrix(:,2));
19 -    eps3 = eps([1;0;0],qMatrix(:,3));
20 -    eps4 = eps([1;0;0],qMatrix(:,4));
21 -    exp1 = exp([0;0;1],theta1,qMatrix(:,1));
22 -    exp2 = exp([1;0;0],theta2,qMatrix(:,2));
23 -    exp3 = exp([1;0;0],(theta3-theta2),qMatrix(:,3));
24 -    exp4 = exp([1;0;0],-theta3,qMatrix(:,4));
25 -    g_ab0 = [eye(3),[0;1.505;1.470];0,0,0,1];
26 -    g_ab = exp1 * exp2 * exp3 * exp4 * g_ab0
27
28      %% Q2----------------------------------------------------------------------
29 -    eps2_t = Ad(exp1)*eps2;
30 -    eps3_t = Ad(exp1*exp2)*eps3;
31 -    eps4_t = Ad(exp1*exp2*exp3)*eps4;
32 -    J = [eps1,(eps2_t-eps3_t),(eps3_t-eps4_t)];
33 -    V = J * [theta1_dot; theta2_dot; theta3_dot];
34 -    V_hat = E_H(V);
35 -    position = [p(g_ab);1];
36 -    v_ab = V_hat * position
37
命令行窗口

v_ab =

   1.1298
  -0.5290
  -1.6356
        0
```

Figure 10: Calculation result for Task 2

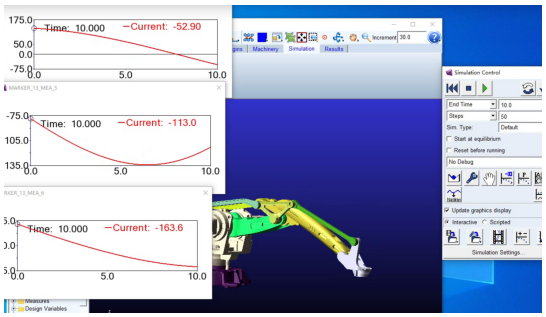

Figure 11: Verification for Task 2

```
ans =

                                                0
  -2.367957899312599879128215619089 5
  -886.531110381732423231558246208 21
```

Figure 12: Calculation result for Task 3

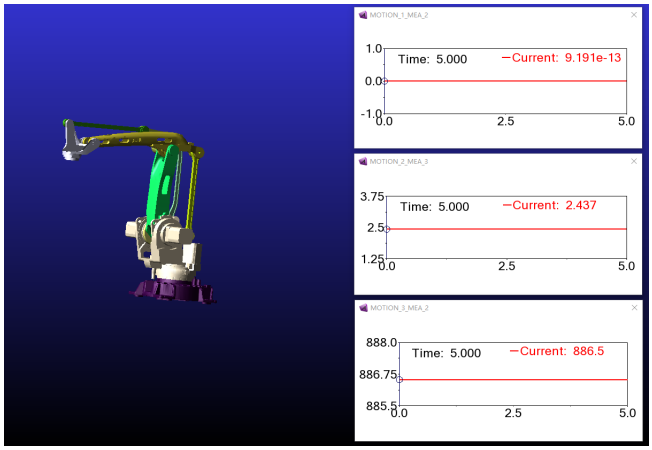

Figure 13: Verification for Task 3

```
ans =

                                                  0
 -73.653528100281749710073120984 75
 66.496615842571114284907389119375
```

Figure 14: Calculation result for Task 4

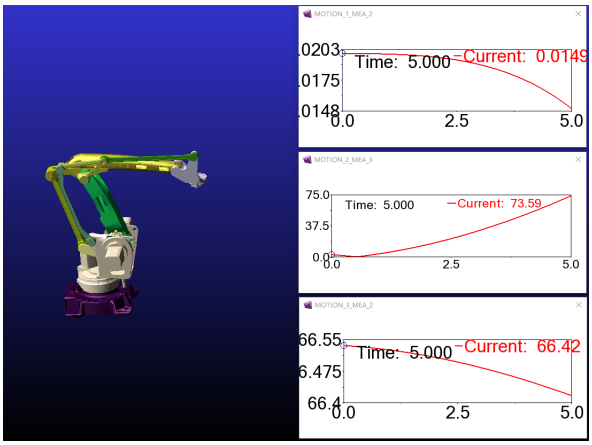

Figure 15: Verification for Task 4

Ownership

		Work distributions			
member	task1 (%)	task2 (%)	task3 (%)	task4 (%)	task5 (%)
LI Xinjie	0	20	100	0	0
ZHANG Hengbin	0	0	0	0	100
XIAO Yunzhong	0	0	0	90	0
MO Zhongtian	0	0	0	0	0
HE Buwei	100	80	0	10	0

Figure 16: Ownership