Contents

1 Probelm

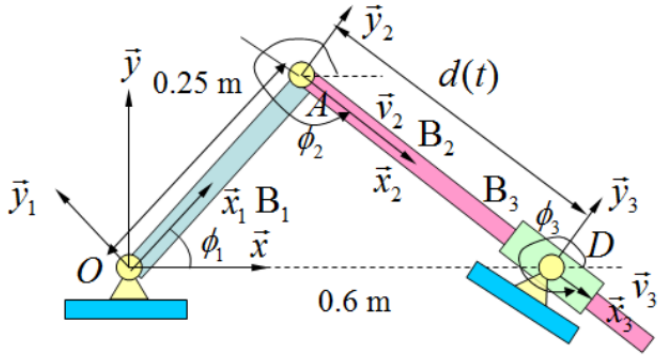

Figure 1: Problem Geometry

Assuming that the length of the crank B_1 is 0.25, and the distance between O and D is 0.6, the system is driven by $d(t) = 0.65 - 0.1t$. Write the constraint equations. Write the Jacobian matrix and right side of acceleration constraint equation. Give the curve of $x_2(t)$, $\phi_2(t)$, $\dot{x_2}(t)$, $\dot{\phi}_2(t)$, $\ddot{x}_t(t)$, $\ddot{\phi}_2(t)$, and give the animation of the motion for $0 \leq t \leq 2.5s$.

2 Solution

2.1 Table of Kinematic Constraints and Driving Constraints

H_i		B_j			B_i			
i	kind	j	$s_j'^{QT}$	$v_j'^{QT}$	i	$s_i'^{QT}$	$v_i'^{QT}$	c
1	(ax)				1	0 0		0
2	(ay)				1	0 0		0
3	(r)	2	0 0		1	0.25 0		
4	(t)	2	0 0	1 0	1	0 0	1 0	
5	(ax)				3	0 0		0.6
6	(ay)				3	0 0		0
7	(tdd)	2	0 0	1 0	3	0 0		$d(t)$

Figure 2: Constraints Table

The input in the code is as follows.

```
paras=[\
['ax', 0, np.array([[0],[0]]), np.zeros((2,1)), 1, np.zeros((2,1)), np.zeros((2,1)), 0, 0], \
['ay', 0, np.array([[0], [0]]), np.zeros((2, 1)), 1, np.zeros((2, 1)), np.zeros((2, 1)), 0, 0], \
['r', 2, np.zeros((2,1)), np.zeros((2, 1)), 1, np.array([[0.25],[0]]), np.zeros((2, 1)), 0, 0], \
['t',2,np.zeros((2,1)),np.array([[1],[0]]),3,np.zeros((2,1)),np.array([[1],[0]]), 0, 0], \
['ax', 0, np.array([[0],[0]]), np.zeros((2,1)), 3, np.zeros((2,1)), np.zeros((2,1)), 0.6, 0], \
['ay', 0, np.array([[0],[0]]), np.zeros((2,1)), 3, np.zeros((2,1)), np.zeros((2,1)), 0, 0], \
['tdd',2,np.zeros((2,1)),np.array([[1],[0]]),3,np.zeros((2,1)),np.zeros((2,1)), 0, 0] \
]
```

Figure 3: The Input Table

2.2 Constriant Equations

$$\boldsymbol{\Phi}=\begin{bmatrix}\boldsymbol{\Phi}^{\boldsymbol{K}}\\ \boldsymbol{\Phi}^{\boldsymbol{D}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{x}^T\boldsymbol{r}_1\\ \boldsymbol{y}^T\boldsymbol{r}_1\\ \boldsymbol{r}_2-\boldsymbol{r}_1-\boldsymbol{A}_1\begin{bmatrix}0.25 & 0\end{bmatrix}^T\\ \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{B}_2{}^T(\boldsymbol{r}_3-\boldsymbol{r}_2)\\ -\begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{B}_{23}\begin{bmatrix}1\\0\end{bmatrix}\\ \boldsymbol{x}^T\boldsymbol{r}_3-0.6\\ \boldsymbol{y}^T\boldsymbol{r}_3\\ \begin{bmatrix}1 & 0\end{bmatrix}(\boldsymbol{r}_3-\boldsymbol{r}_2)-(0.65-0.1t)\end{bmatrix}=\boldsymbol{0} \tag{1}$$

2.3 Jacobian

To calculate the Jacobian, some parameters are given below:

$$\dot{\boldsymbol{A}}_1=\boldsymbol{B}_1\dot{\phi}_1,\dot{\boldsymbol{B}}_3=-\boldsymbol{A}_3\dot{\phi}_3 \tag{2}$$

$$\dot{\boldsymbol{B}}_{23}=\frac{\partial(\boldsymbol{R}\boldsymbol{A}_{23})}{\partial(\phi_3-\phi_2)}(\dot{\phi}_3-\dot{\phi}_2)=\boldsymbol{R}\boldsymbol{B}_23(\dot{\phi}_3-\dot{\phi}_2)=\boldsymbol{R}\boldsymbol{R}\boldsymbol{A}_{23}(\dot{\phi}_3-\dot{\phi}_2)=-\boldsymbol{A}_{23}(\dot{\phi}_3-\dot{\phi}_2) \tag{3}$$

Calculate the Jacobian manually

$$\boldsymbol{\Phi}_q = \begin{bmatrix} \boldsymbol{x}^T & 0 & \mathbf{0}_{1\times2} & 0 & \mathbf{0}_{1\times2} & 0 \\ \boldsymbol{y}^T & 0 & \mathbf{0}_{1\times2} & 0 & \mathbf{0}_{1\times2} & 0 \\ -\boldsymbol{I}_{2\times2} & -\boldsymbol{B}_1\begin{bmatrix}0.25\\0\end{bmatrix} & \boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{1\times2} & 0 & -\begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{B}_2^T & -\begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{A}_2^T(\boldsymbol{r}_3-\boldsymbol{r}_2) & \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{B}_2^T & 0 \\ \mathbf{0}_{1\times2} & 0 & \mathbf{0}_{1\times2} & -\begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{A}_{23}\begin{bmatrix}1\\0\end{bmatrix} & \mathbf{0}_{1\times2} & \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{A}_{23}\begin{bmatrix}1\\0\end{bmatrix} \\ \mathbf{0}_{1\times2} & 0 & \mathbf{0}_{1\times2} & 0 & \boldsymbol{x}^T & 0 \\ \mathbf{0}_{1\times2} & 0 & \mathbf{0}_{1\times2} & 0 & \boldsymbol{y}^T & 0 \\ \mathbf{0}_{1\times2} & 0 & -\begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{A}_2^T & \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{B}_2^T(\boldsymbol{r}_3-\boldsymbol{r}_2) & \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{A}_2^T & 0 \end{bmatrix} \tag{4}$$

2.4 Right Side of Velocity Equations

The right side of velocity equations is as follows:

$$\boldsymbol{v} = \begin{bmatrix} \mathbf{0}_{8\times1} \\ -0.1 \end{bmatrix} \tag{5}$$

2.5 Right Side of Acceleration Equation

The right side of acceleration equation can be calculated as the following formula[1]:

$$\boldsymbol{\gamma} = -(\boldsymbol{\Phi}_q\dot{\boldsymbol{q}})_q\dot{\boldsymbol{q}} - 2\boldsymbol{\Phi}_{\boldsymbol{q}t}\dot{\boldsymbol{q}} - \boldsymbol{\Phi}_{tt} \tag{6}$$

The right side of acceleration is as follows:

$$\boldsymbol{\gamma} = \begin{bmatrix} 0 \\ 0 \\ -\boldsymbol{A}_1 \begin{bmatrix} 0.25 & 0 \end{bmatrix}^T \dot{\phi}_1^2 \\ 2 \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{A}_2^T \dot{\phi}_2 (\dot{\boldsymbol{r}}_3 - \dot{\boldsymbol{r}}_2) + \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{B}_2^T (\boldsymbol{r}_3 - \boldsymbol{r}_2) \dot{\phi}_2^2 \\ - \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{B}_{23} \begin{bmatrix} 1 \\ 0 \end{bmatrix} (\dot{\phi}_3 - \dot{\phi}_2)^2 \\ 0 \\ 0 \\ -2 \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{B}_2^T \dot{\phi}_2 (\dot{\boldsymbol{r}}_3 - \dot{\boldsymbol{r}}_2) + \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{A}_2^T (\boldsymbol{r}_3 - \boldsymbol{r}_2) \dot{\phi}_2^2 \end{bmatrix} \tag{7}$$

3 Brief Introduction of the Program

All the above results are obtained by manual calculation. Now it comes to a brief introduction of my program.

3.1 Programming Environment

Considering the program's implementability, code's readability and users' friendliness, python will be a great option for it ranks top in all the coding language and it's easy to understand for the beginner on the Coding.

Here is the programming version environment:

- python 3.8 (python3 will be OK)
- numpy 1.20.1
- matplotlib 3.3.4
- imageio 2.9.0
- PIL 8.2.0

3.2 Program Structure

The total program structure is simple. In the Code directory there is a file main.py, which is the entry of the main program, and a library directory kinematics, where all the classes and the function are stored. What the files in the kinematics for is as follows:

- **Constraints.py**

  It contains all the constraints in the kinematics, including how to calculate the jacobian, right side of the velocity equation and the right side of the acceleration equation of each constraints. It can be developed into a General library, what I have achieved in the project, which means it includes all the constraints mentioned in the class.

- **Drives.py**

  The same as the Constraints.py, it contains all the drive constraints in the kinematic and the parameters can be changed in the file.

- **Kinematics.py**

  It contains a class Kinematics which is used to get the table parameters and solve the kinematic problem, also including the plot function.

- **Animate.py**

  It contains a function to plot the animation of this problem, but not for general problem.

4 Results

The initial position is showed in the following image.

```
'''
q: [x1 y1 phi1 x2 y2 phi2 x3 y3 phi3]
'''

q0 = np.array([[0],[0],[np.pi/2],[0],[0.25],[2*np.pi-np.arctan(5/12)],[0.6],[0],[2*np.pi-np.arctan(5/12)]])
```

Figure 4: Initial Position

Then calculate the results of the motion using the programs. Here, according to the demanding, only the curves of $x_2(t)$, $\phi_2(t)$, $\dot{x}_2(t)$, $\dot{\phi}_2(t)$, $\ddot{x}_t(t)$, $\ddot{\phi}_2(t)$ are given.

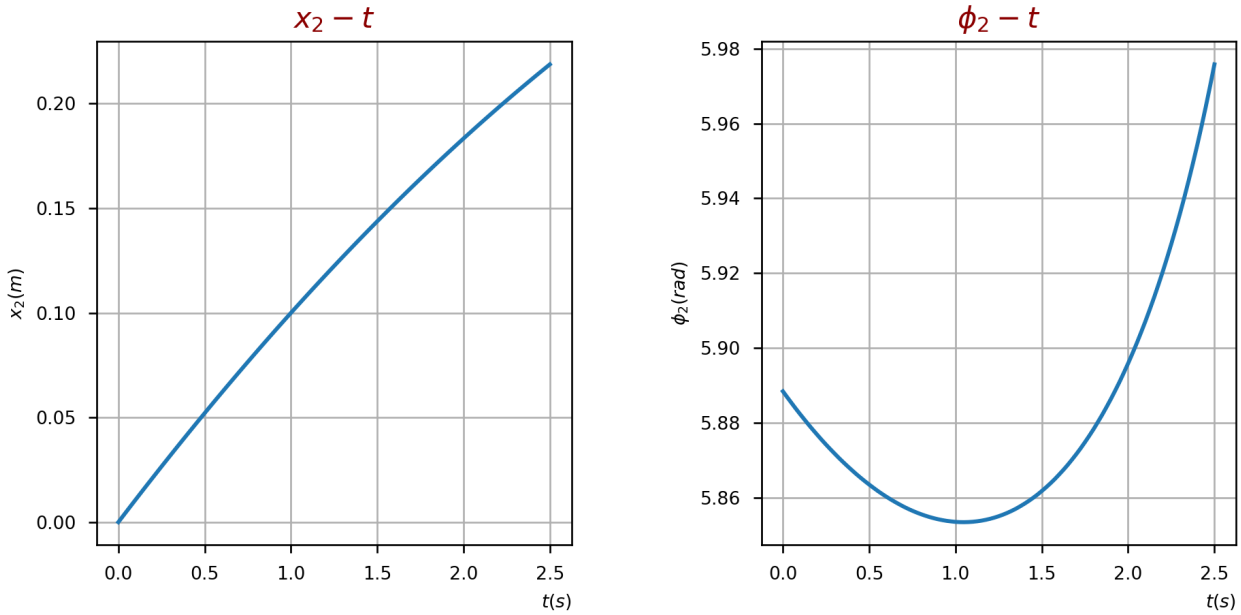

Figure 5: The Curves of $x_2(t)$, $\phi_2(t)$

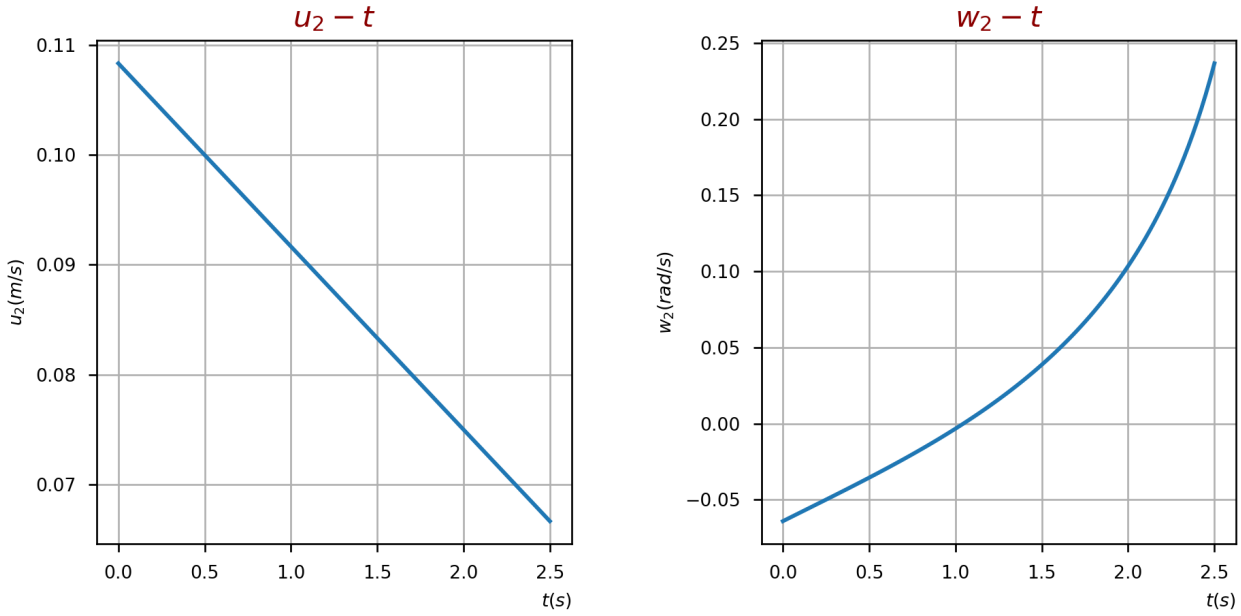

Figure 6: The Curves of $\dot{x}_2(t)$, $\dot{\phi}_2(t)$

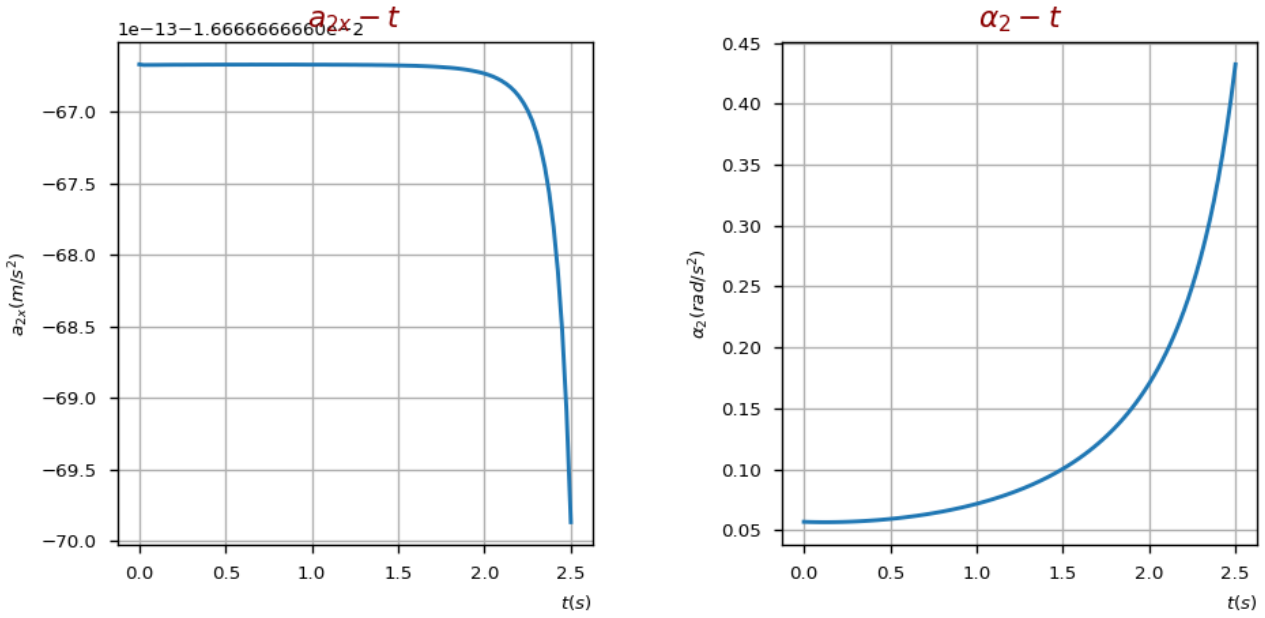

Figure 7: The Curves of $\ddot{x}_2(t)$, $\ddot{\phi}_2(t)$

To get it more clearly, change the coordinate of y of the image of $\ddot{x}_2(t)$.

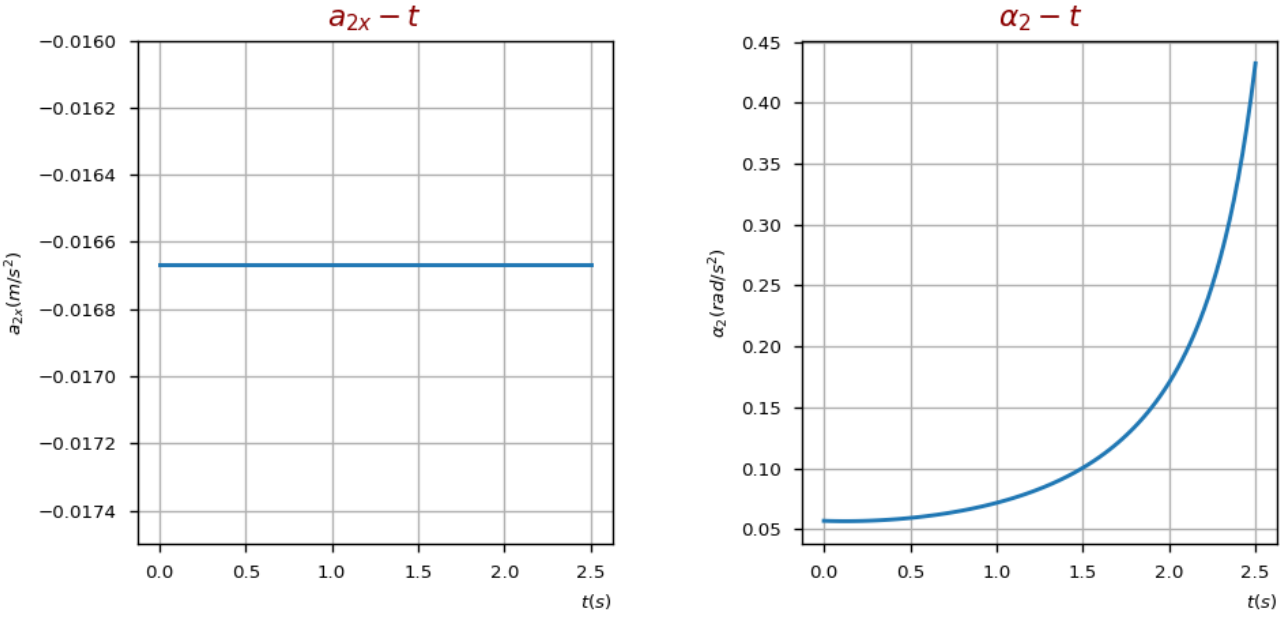

Figure 8: The Curves of $\ddot{x}_2(t)$, $\ddot{\phi}_2(t)$

Furthermore, an animation of the motion is given to decipt the problem more specific by the function of Animation, which is implemented by the matplotlib.animation. The gif file(roll.gif) can be found in the project directory because the file format is not usable in the pdf(which is created by latex directly so I couldn't give the docs version)

5 Analysis

5.1 The Influence of the Drive Velocity

In order to explore the influence of different drive velocities on the motion, three different velocity are chosen in calculation of position, velocity and acceleration, and the drives can be described as the following formula respectively:

$$d_1(t) = 0.065 - 0.08t \tag{8}$$

$$d_2(t) = 0.065 - 0.1t \tag{9}$$

$$d_3(t) = 0.065 - 0.12t \tag{10}$$

The results are shown below:

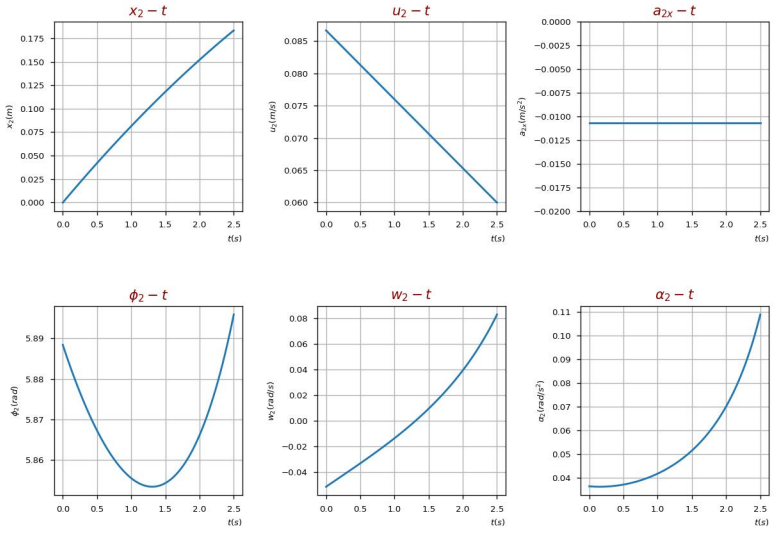

Figure 9: The Motion Curves of d_1

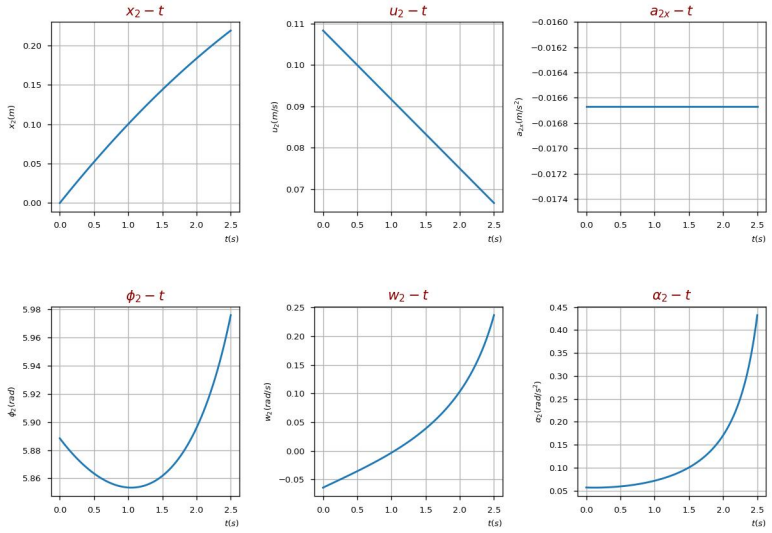

Figure 10: The Motion Curves of d_2

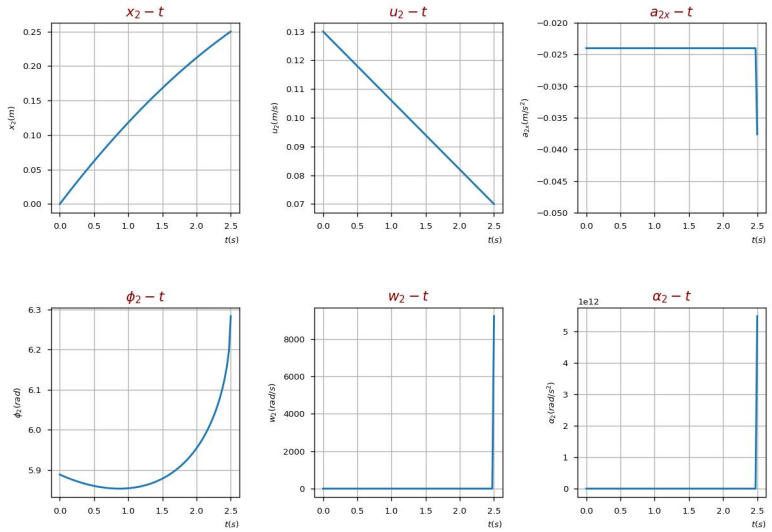

Figure 11: The Motion Curves of d_3

Putting the results together will be more explicit to understand the influence.

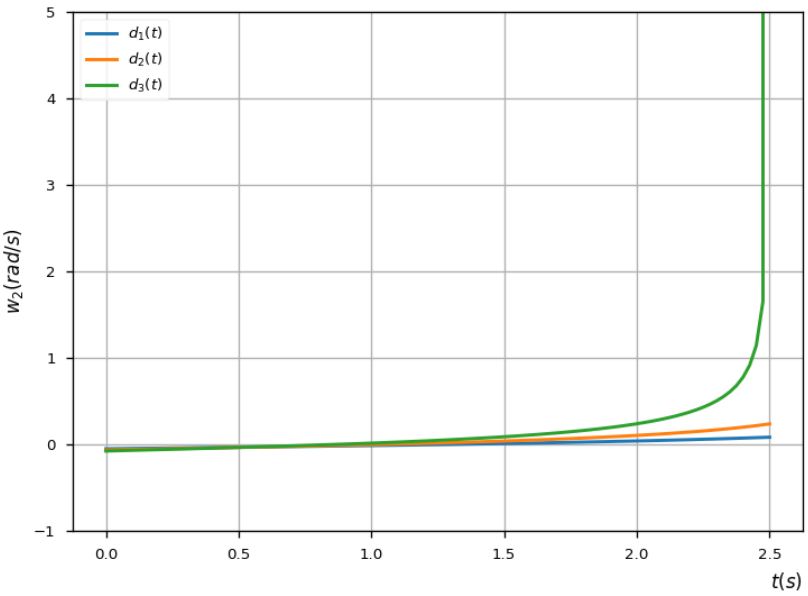

Figure 12: Three Curves of w_2

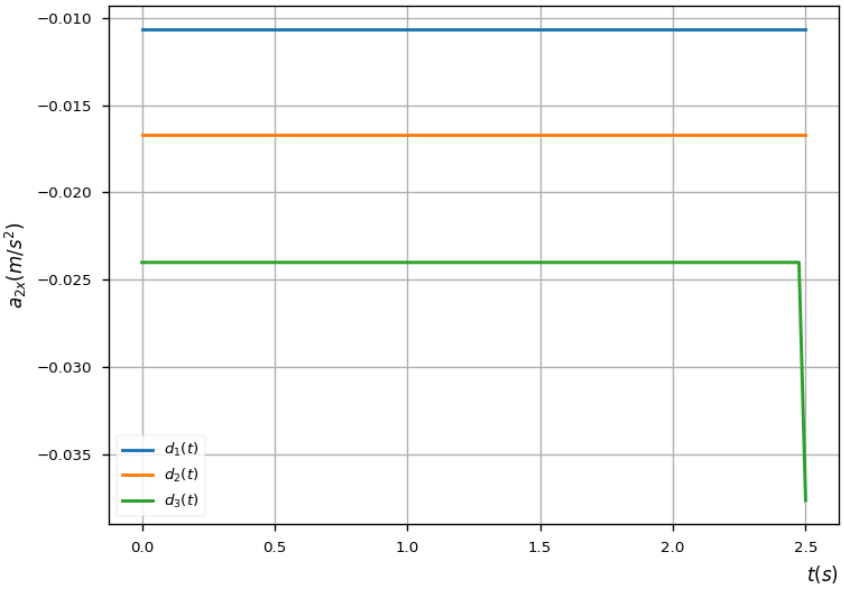

Figure 13: Three Curves of a_{2x}

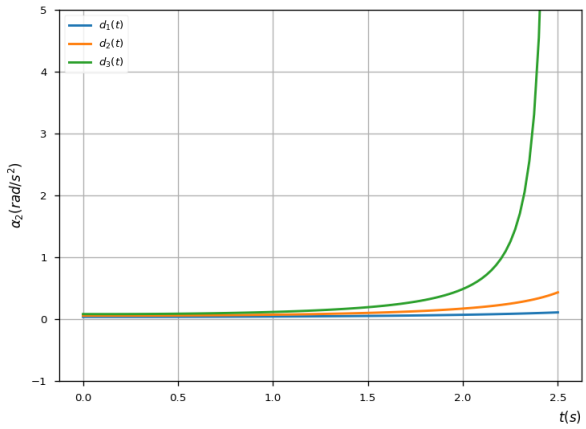

Figure 14: Three Curves of α_2

According to the figures above, from $0.08m/s$ to $0.12m/s$, with the growth of the drive velocity, the drive, position and the acceleration of the body 2 are becoming more and more sensitive about the change of time. And with the development of the time, the velocity and acceleration will trend to singular value when time is trending to 2.5s, which means they will become extremely large.

5.2 Mathematics on the Singularity

When $v = 0.12m/s$ and $t = 2.5s$, $q = \begin{bmatrix} 0 & 0 & 0 & 0.25 & 0 & 0 & 0.6 & 0 & 0 \end{bmatrix}^T$, and the Jacobian and right side of the velocity is as follows:

$$\mathbf{\Phi}_q = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -0.25 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -0.35 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{11}$$

$$\boldsymbol{v} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -0.12 \end{bmatrix} \tag{12}$$

Since $|\boldsymbol{\Phi}_q| = 0$, there is a singularity happends. To determine the type of the singular configuration, we need to solve the velocity equation[2].

$$\boldsymbol{\Phi}_q \dot{\boldsymbol{q}} = \boldsymbol{v} \tag{13}$$

Note that rank of $\boldsymbol{\Phi}_q$ is 8, and the rank of augmented matrix $\begin{bmatrix} \boldsymbol{\Phi}_q & \boldsymbol{v} \end{bmatrix}$ is 9, so the velocity equation has no solution[3]. Therefore the locked up happends when $v = 0.12m/s$ and $t = 0.25s$.

In conclusion, we could find that when the drive velocity is $0.12m/s$ at the time $2.5s$, locked up appears. In the figure, it shows that v, w, a, α will become extremely large at the time $2.5s$, and the motion can not continue when the locked up happend. Therefore, the singularity should be avoided in the mechanism.

5.3 Advantages and Disadvantages

For the advantage of the project, I achieved the universal solution for the kinematics problems here by including all the constraints mentioned in the class.Moreover, because it's developed by python, it's more portable and have more development space than the project based on matlab.

As for the flaws of the project, considering that the python could do the symbolic operation as simple as matlab, the drive library is developed by using the same drive expression, so not such universal at this point. And the animation function is designed for this specific problem, and its universality is still in my consideration.

References

[1] Edward J Haug. *Computer aided kinematics and dynamics of mechanical systems. Vol. 1: basic methods*. Allyn & Bacon, Inc., 1989.

[2] Parviz E Nikravesh. *Planar Multibody Dynamics: Formulation, Programming with MATLAB®, and Applications*. CRC Press, 2018.

[3] Gilbert Strang. *Linear algebra and its applications.* 2006.