# **Weekly Report – W11 Fall 2022**

## **Problem**

1. Check the inertia matrix of falling SRA model.

## **Solution**

1. Inertia matrix verification

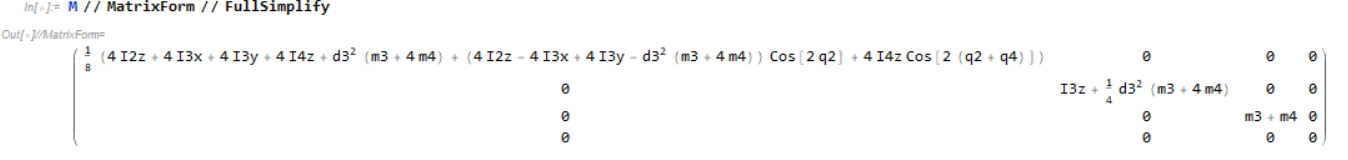
#### (1). Derivations from other papers

As all the governing equations have been derived by Mathematica last week, and they were plugged into the Simulink function block to perform the simulation, however, after fixing all the formatting issues of importing expressions from Mathematica to MATLAB/Simulink, the debugger indicated that there was singularity when , the step size and tolerance error were suggested to be set smaller. After multiple times of trials, it still didn’t work, so I turned to use one of the implicit ODE solvers ode15s to attempt to obtain some simulation results, it failed again. The reason behind was summarized as follows:

* Singularity mostly comes from the inertia matrix, since the Coriolis-centripetal (short for C in the following paragraph) matrix is derived on basis of the inertia matrix, if there are some issues with the C matrix, the origin of the problem can be traced back to inertia (M) matrix. Besides, singularity means that the specific matrix is invertible, further more, that is to say the matrix is not positive definite, which indicates that the inertia matrix derived is not correct;
* The inertia matrix contains so many high order terms like , which could be a sign that the derivation is wrong.

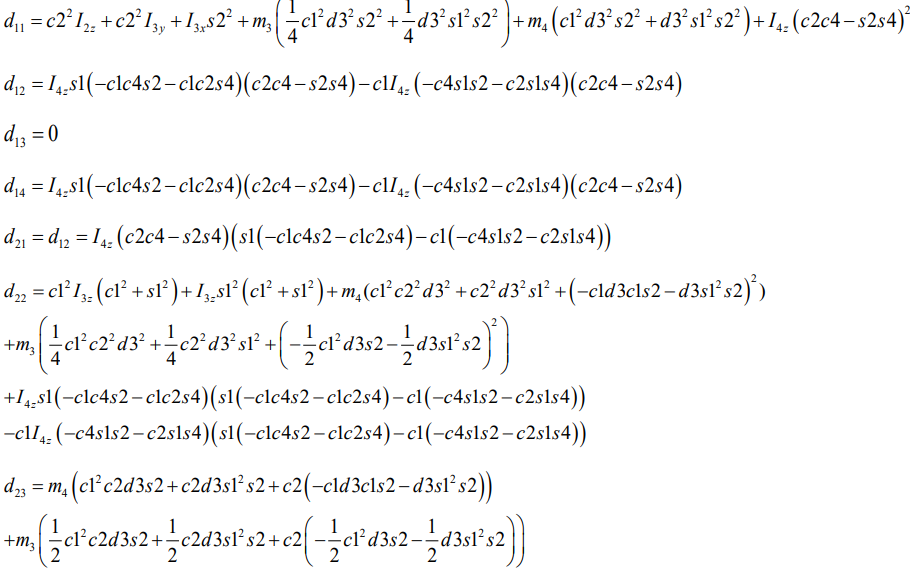
Since the falling SRA model is established based on the Piesewise Constant Curvature (PCC) theory, the basic principles should be the same though the establishment of the coordinate system might be slightly different, so the possible solution is to find if there exist some papers containing explicit derivation process of the inertia matrix at least. Ian [1] has explicitly state the derivation process of a single segment, the only difference from my version is there are no high order terms for the denominator for the elements in the inertia matrix. To evaluate it, we start from the rotational matrix of the end-effector,

After deriving in Mathematica by fully simplification, the inertia matrix will be



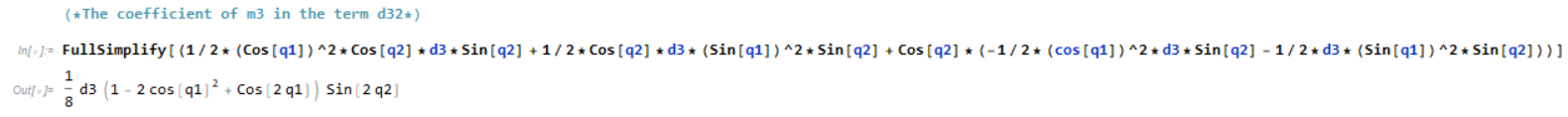
**Fig. W11-1** The inertia matrix derived by Mathematica

However, most of elements were totally different from the continuum robot virtual RRRR robot dynamics derivation, as none of them have been fully simplified using Mathematica, part of expressions of elements in inertia matrix in Ian’s paper will be shown as follows,



**Fig. W11-2** The derivation of elements in inertia matrix in Ian’s paper

After computing these elements, all the terms were equal to what I have derived except for the element , which is showcased as follows,



**Fig. W11-3** The element in inertia matrix which cannot be fully simplified by whatever means

Obviously we can see that the term should be equal to zero, in this case, the inertia matrix derived is proved to be correct. But there is still one thing worth to note that sometimes Mathematica is not so reliable, it would not achieve what we desire to see all the time; so back to the derivation of falling SRA, though the expression of inertia matrix contains so many high order terms, it is still possible to be correct due to the performance of Mathematica (not a strong standing though).

Besides, the main reason for no terms such as existing in Ian’s paper lies in that the chord length was not substitute by the bending angle . After doing that, we can find that the term will frequently appear in the denominator in the inertia matrix. And since the derivation is just for single segment in Ian’ paper, for our project, the falling SRA is composed of two equal-length segments, which means that there will be some terms like for the denominator in the inertia matrix owing to the multiplication of the rotational matrix, which will be the second indicator that my derivation could be correct.

### (2). Derivation for falling SRA

Since my understanding for deriving the inertia matrix (or the governing equation) is proved to be correct, the next step is to compare the derivation results of using the definition (Jacobian matrix) and my original version (slightly different from definition, the linear velocity Jacobian was derived by differentiating the first three elements of the last column in the rotation matrix).

To examine if the two matrices are exactly the same, substration has been made, the result is not zero, the possible reasons are as follows:

* The FullSimplify function in Mathematica could not guarantee a completely simplified version of a certain expression, just like what has been shown in Fig. W11-3;
* The way to derive the inertia tensor matrix, matrix is different, I have to make a double check in the next week.

My concern is that, if the inertia matrix can be proved to be positive definite, but there still exists singularity at the beginning of the simulation, are there any other reasons for that? Maybe I can do some follow up study.

## **Plan**

1. Examination of the inertia matrix;
2. Identify what kind of ODE solver has been used in Chase’s model, if it hasn’t been specified, find out if it has been mentioned in other related papers;
3. Self-study about energy shaping and PDE solving;
4. Modifications about the part “back thigh panel” of the exoskeleton.

## **Reference**

[1] Wang, C. *et al.* (2021) “Dynamic control of multisection three-dimensional continuum manipulators based on virtual discrete-jointed robot models,” *IEEE/ASME Transactions on Mechatronics*, 26(2), pp. 777–788. Available at: https://doi.org/10.1109/tmech.2020.2999847.