**Weekly Report – W18 Fall 2022**

**Problem & Task**

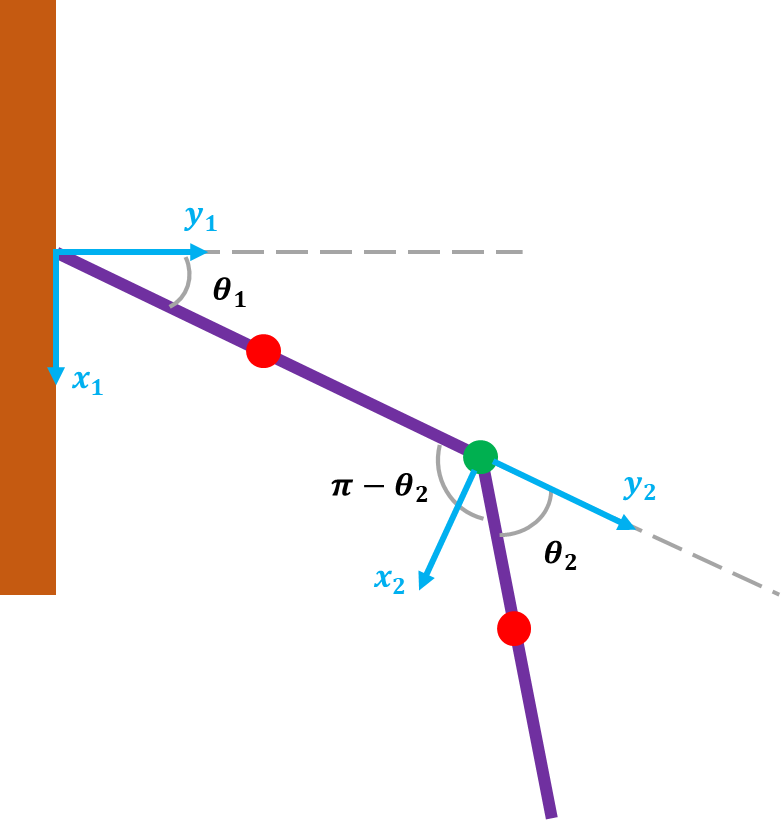
1. Continue to use MATLAB to model the falling ball tied with SRA scenario;
2. Update about the remained problems proposed several weeks ago about the TMTDyn package, since this week we had a direct communication with the author, the key problems have been solved, more details are shown below.

**Solution**

1. *Update about the “Falling SRA” project*

(1). Equations of motion derivations

To derive the governing equations of the system, we firstly have to define the degree of freedom for it, as shown in the figure below, we can draw a dash line which is perpendicular to the wall, and it is deemed to be the y axle of the first frame (for short, we will use F+No. to represent different frame or coordinate systems in the subsequent paragraph), and accordingly, the axle pointing to the ground which is perpendicular to the y axle will be the x axle for F1. And the rule for establishing the following frames will be the same.



**Fig. W18-1** The schematic of the definition of the degree of freedom for one of the SRAs

For deriving the rotation matrix of each joint, rather than using D-H convention (take the z axle always as the rotation axle or the prismatic operation direction), we are going to directly operate about the original axis system, if we assume the origin is located at the center of the two walls on the ground (at the same time, the height of the fixed point is 1 m and the distance between two walls is 1 m), the computed rotation matrices of each joint should be as follows:

Left:

Right:

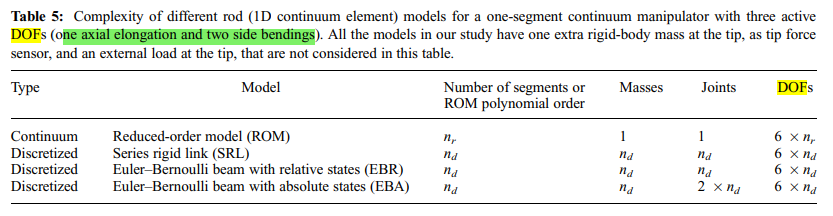
However, when using the information in those rotation matrices to work out the Jacobian matrix for each link, the MATLAB code suffered from some errors because the value of type “sym” is not convertible to “double”, I’m still trying to find a solution to it, all the work I have done is to make my code more adaptable for more flexible number of links settings, I will fix this problem as soon as possible.

1. *Minutes of meeting in terms of the remained questions of TMTDyn*

This week I had a meeting with the author of TMTDyn package, Dr. Sadati, though the meeting didn’t last so long (just about one and a half hours), I think we have solved many concerns. And Dr. Sadati is a very nice person, he gave me lots of suggestions on simulation, and we have set up contact in the long time, which will be convenient for me to ask more questions in the future. And our discussion is summarized as follows:

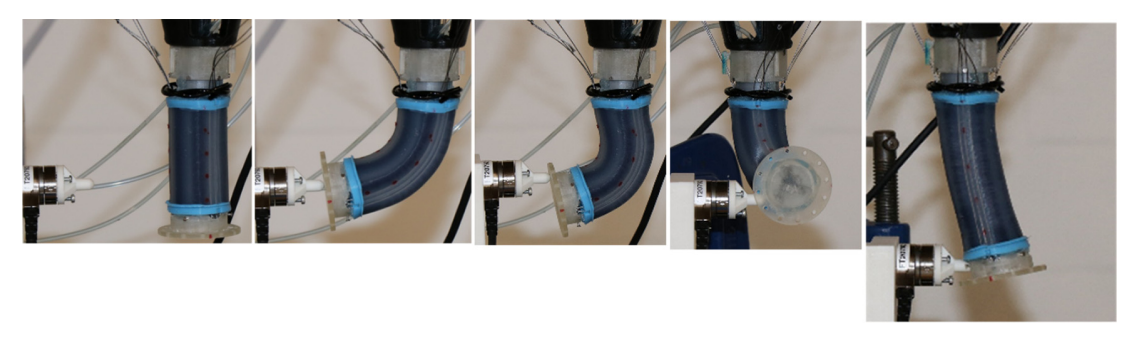
(1). Question:

For the examples Sadati provided in the latest version on GitHub, more specifically, for exp2\_EBR, when I changed the number of segments from 2 to 3, I found that the size of the output results (they should be the intermediate variables of the ODE solvers, like positions and velocities) varied from 2801\*24 to 2801\*36 when exp\_case=2, which means that by increasing each segment/link, there will be another 12 variables, if we assume the variables are composed of positions and velocities (not including accelerations), there will be 6 DoFs for each link. So I'm a little confused that in the paper "*TMTDyn: A Matlab package for modeling and control of hybrid rigid–continuum robots based on discretized lumped systems and reduced-order models*", there are only three DoFs introduced, which are one axial elongation and two side bendings respectively. But in the table below (still in this paper), there are 6 DoFs for each segment, so what will be the rest three?



**Fig. W18-2** The screen shot from the author’s paper about definition of DoF

Answer:

The author admitted that the description here in the paper is little confusing and vague, but the demonstration in the title of the table and the content below are not ambivalent, 

**Fig. W18-3** The schematic of the SRA motion defined in the author’s paper

as explained by Dr. Sadati, the three DoFs were just established based on the most simple case, 2D problem as shown in the figure above, the last figure is an example of one axial elongation, the third and the fourth ones are examples of two side bendings, one is bending to the left and right in the paper plane, the other is bending outside and inside of the paper. However, for real SRAs, apparently there are much more needed to be considered, such as the torsion, so another three DoFs are twist angles about the three axles.

In the simulation result, we found that by increasing each SRA link, there will be another 12 variables emerged, which was defined as

According to the formatting of the output data, we can judge that for each link/segment, the first 6 variables are positions (temporarily we are not sure they are the joints’ or the COMs’), and the rest 6 will be the velocities.

(2). Question:

When I set the length of the manipulator to 150 mm (all the other parameters remain the same), the simulation results seemed a little abnormal, the position of the tip can reach about 18,000 m and animation was composed of several rotating frame rather than the cylinders. I guess the reason behind should be that the manipulator was designed to be too "thin", so it did not have the "ability" to swing back to its equilibrium point after the force exerting on the tip is eliminated; after I enlarged the outer diameter or the pressure in the chamber, this issue has gone. It's interesting, I like doing these simulations, and I would appreciate it if you have some comments and recommendations on modelling very "long" manipulators? Also is there any limit for the pressure and force applied (for example, the maximum pressure the chamber can resist) for the type of the manipulator you used in the paper?

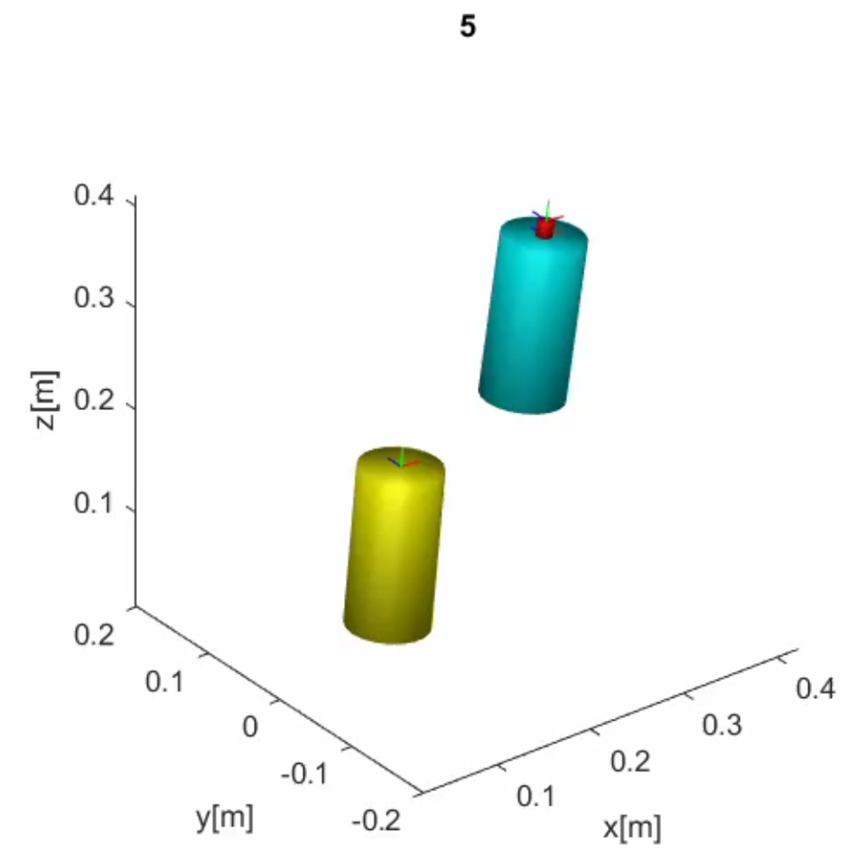
Answer:

Dr. Sadati drew the same conclusion as mine, he also thought:

* The geometry factors domain the final simulation results;
* If the simulation results seemed pretty weird, we can try different solvers, like radau and radau\_mex rather than MATLAB ode solvers; sometimes it might be due to the small step size that the results got crack, we can also change it to lead to a better result;
* There are no limits for the pressure or the force input for the simulation to guarantee a stable result, just because it is simulation, we can try to modify the parameters within a very large range.

(3). Question (about the code itself):

* The simulation result, more specifically, the SRA tip position is partially composed of the physical experiment results from measurement, we don’t need them any more for any other scenarios since it has lost the value to make a comparison, how can we eliminate them?
* After modifying all the geometry factors with our project’s, I found the animation is abnormal, the segments are separated from each other, which can be seen in the example below, what can we do to avoid this issue?



**Fig. W18-4** The example of segments separated from each other

Answer:

* The plot function to draw the physical experimental data is actually located in the path tmtdyn\eom\post\_proc.m, which is the post processing file, so I perfectly missed the target file in the checking work last week;
* The segment separating phenomenon is probably due to the settings of Young’s Modulus and the density(I’ve only changed the density and found that it didn’t affect so much); also because the hardware used in the author’s paper is totally different from our project’s, his SRA has 6 chambers, and ours only has three, so the definition about the vacuum volume is different, even a small variation of pressure input would lead to a large difference (I think it’s the main reason, I will take a try next week).

(4). Question:

In the future, I think I need to simulate for impact dynamics of soft robot as well, so any suggestion about that? I think the main challenge is that after the impact interfered, the dynamics of the whole system is not continuous, we have to treat the impact process individually.

Answer:

The author hasn’t added such function in the package so far, and actually it has been two years that the author hasn’t modified the package, his colleagues have been working on it recent years, he will consider about my idea. Also he gave me some suggestions on it, as I thought, he also suggested the impact process can be separated into 3 parts, before, after and during the impact instant (very short time period), and we assume during the impact stage, the deformation is so small that it can be ignored, so to work out the velocity of the contact point before and after the contact instant is the key; and the instant we can modify the inertia matrix individually, then plug the velocity into the next stage, which is also continuous as before the impact. The logic is basically the same as mine, and feasibility of coding has been further proved by the author, which is a good sign for our simulation and research in the future.

**Plan**

1. Try to do the simulation according to the suggestions from Dr. Sadati, I think I have quite a lot to test and make a comparison, the target by the end of next week should be eliminating the physical experimental data and solving the segment separating problem at least.