

University of New Mexico

## UNM Digital Repository

---

Mechanical Engineering ETDs

Engineering ETDs

---

Winter 11-11-2019

# Development of MBD Model and Analysis Methodology for Vehicle Design

Alfonso Christopher Ponce  
*University of New Mexico*

Follow this and additional works at: [https://digitalrepository.unm.edu/me\\_etds](https://digitalrepository.unm.edu/me_etds)



Part of the [Automotive Engineering Commons](#), [Computer-Aided Engineering and Design Commons](#), and the [Dynamics and Dynamical Systems Commons](#)

---

### Recommended Citation

Ponce, Alfonso Christopher. "Development of MBD Model and Analysis Methodology for Vehicle Design." (2019). [https://digitalrepository.unm.edu/me\\_etds/176](https://digitalrepository.unm.edu/me_etds/176)

This Thesis is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Mechanical Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact [amywinter@unm.edu](mailto:amywinter@unm.edu), [lsloane@salud.unm.edu](mailto:lsloane@salud.unm.edu), [sarahrk@unm.edu](mailto:sarahrk@unm.edu).

Alfonso Christopher Ponce

*Candidate*

Mechanical Engineering

*Department*

This thesis is approved, and it is acceptable in quality and form for publication:

*Approved by the Thesis Committee:*

Dr. John Russell, Chairperson

Dr. Yu-Lin Shen

Dr. Richard Jepsen

**DEVELOPMENT OF MBD MODEL AND ANALYSIS  
METHODOLOGY FOR VEHICLE DESIGN**

by

**ALFONSO CHRISTOPHER PONCE**

**BACHERLORS OF SCIENCE MECHANICAL ENGINEERING**

THESIS

Submitted in Partial Fulfillment of the  
Requirements for the Degree of

**Master of Science  
Mechanical Engineering**

The University of New Mexico  
Albuquerque, New Mexico

**December, 2019**

## **ACKNOWLEDGEMENTS**

I want to acknowledge Dr. John Russell for the opportunity to learn from teaching and grading. For acting as my chair and for the amazing support and guidance through this paper. A mentor and scholar I hope to be like one day.

I also like to thank Dr. Yu-Lin Shen and Dr. Richard Jepsen for their time as my acting committee in such short notice and flexibility.

To my mentor Luis, for teaching me patience in learning and to remain humble, through anything and everything.

To Mariah, Tinks, and Ghost for all your patience and support, especially when things were at the bleakest. This could have not been possible without you.

To my family, words cannot express everything you have done for me, that your love and support has helped me push through any obstacle. Thank you

# **Development of MBD Model and Analysis Methodology for Vehicle Design**

by

**Alfonso Christopher Ponce**

**B.S., Mechanical Engineering, University of New Mexico, 2017**

**M.S., Mechanical Engineering, University of New Mexico, 2019**

## **ABSTRACT**

This thesis is to provide a new methodology that considers the effects the chassis applies during dynamic load to the suspension and the handling characteristics of a road course vehicle. A methodology in which is described in detail as well as an applied demonstration where literature before this thesis, lacks in replication and serves to establish a basis for multi-body dynamic analysis in this area. Two models to be created, rigid and flexible, and to compare each other to study the effects the chassis torsional stiffness has during load transfer. Ultimately, to assist in the design of a vehicle that takes into account the effects studied in this thesis.

# TABLE OF CONTENTS

<b>LIST OF FIGURES</b> .....	vii
<b>LIST OF TABLES</b> .....	ix
<b>1. INTRODUCTION</b> .....	1
1.1 Background.....	1
1.2 Problem Statement.....	1
1.2.1 Limitations and Defining Measurement of Performance.....	2
1.3 Literature .....	3
1.4 Model Process Development.....	4
1.4.1 Software Selection.....	4
1.4.2 Development.....	7
1.5 Demonstration of Process .....	7
1.5.1 Reference used.....	7
1.5.2 Demonstration Value to Development.....	8
<b>2. MODELING PROCESS</b> .....	9
2.1 Suspension .....	11
2.1.1 Data Compiling.....	11
2.1.2 Building the Template .....	14
2.1.3 Creating the Parts.....	19
2.1.4 Adding Connections .....	22
2.1.5 Suspension verification .....	25
<b>2.2 Chassis</b> .....	28
2.2.1 Chassis Preparations .....	29
2.2.2 Data Compiling.....	29
2.2.3 Import.....	30
2.2.4 Model Merging.....	31
2.2.5 ADAMS/Flex .....	33
<b>2.3 Chassis &amp; Suspension</b> .....	37
<b>3. APPLICATION OF PROCESS</b> .....	43
3.1 Suspension .....	43
3.1.1 Data Compiling.....	44
3.1.2 Building Template .....	45
3.1.3 Creating the Parts.....	49

3.1.4	Adding Connections .....	52
3.1.5	Suspension verification .....	56
3.2	Chassis .....	58
3.2.1	Chassis Preparations .....	59
3.2.2	Data Compiling .....	59
3.2.3	Importing .....	60
3.2.4	Model Merging .....	61
3.2.5	ADAMS/ Flex .....	63
3.3	Suspension and Chassis .....	66
3.3.1	Simulation .....	67
3.3.2	Results & Discussion .....	69
<b>4.</b>	<b>CONCLUSION .....</b>	<b>76</b>
<b>5.</b>	<b>FUTURE WORK .....</b>	<b>76</b>
	<b>APPENDICES .....</b>	<b>80</b>
	<b>Appendix A. Suspension Parameters .....</b>	<b>80</b>
	<b>Appendix B. PART CREATION GUIDE .....</b>	<b>82</b>
	<b>Appendix C. ADAMS 2018 FSAE MODEL STRUCTURE (By Parts) .....</b>	<b>83</b>
	<b>Appendix D. ADAMS 2018 FSAE MODEL STRUCTURE (By Connections) .....</b>	<b>88</b>
	<b>Appendix E. Race Technology/ GoPro Data .....</b>	<b>90</b>
	<b>References .....</b>	<b>94</b>

## LIST OF FIGURES

Figure 1. ADAMS/ Car powertrain parameters .....	5
Figure 2. Methodology work flow diagram.....	10
Figure 3. Reference frame .....	14
Figure 4. Marker Icon .....	16
Figure 5. Table editor .....	17
Figure 6. Parameters .....	19
Figure 7. Solids .....	21
Figure 8. Contact feature.....	24
Figure 9. Tire value.....	24
Figure 10. Force vector feature - model input.....	25
Figure 11. Simulation control window – equilibrium configuration .....	26
Figure 12. Solver setting adjustments .....	28
Figure 13. Import configuration.....	30
Figure 14. Material defining window .....	31
Figure 15. Merge window .....	32
Figure 16. Boolean, merge without contact.....	32
Figure 17. Boolean, merge in contact.....	34
Figure 18. ViewFlex Mesh window .....	35
Figure 19. Mesh iterations.....	36
Figure 20. ViewFlex attachment window .....	37
Figure 21. Lane change diagram .....	38
Figure 22. STEP function defined (5) .....	39
Figure 23. Force vector input window.....	40
Figure 24. Slalom maneuver diagram.....	40
Figure 25. Damped sinusoidal wave example.....	41
Figure 26. 2018 UNM LOBOMotorSports FSAE vehicle .....	43
Figure 27. Front right corner of the 2018 FSAE vehicle .....	44
Figure 28. Rear right corner of the 2018 FSAE vehicle .....	44
Figure 29. Right side of the template created .....	46
Figure 30. Naming Convention used for template .....	47
Figure 31. Mirror function depicted .....	48
Figure 32. CG marker location.....	49
Figure 33. Full vehicle template complete .....	49
Figure 34. Vehicle mass value.....	50
Figure 35. Front suspension parts added .....	51
Figure 36. Rear suspension parts added.....	51
Figure 37. Full suspension- parts added and complete.....	52
Figure 38. Full suspension- joints added .....	53
Figure 39. Front shock setup .....	54
Figure 40. Rear shock setup.....	54
Figure 41. Front ARB torsion spring .....	54
Figure 42. Rear ARB torsion spring.....	55
Figure 43. Tire stiffness and damping values used.....	55



Figure 44. Full suspension- connections added, suspension model completed .....	56
Figure 45. Equilibrium test – full scale.....	56
Figure 46. Equilibrium test – scale of interest .....	57
Figure 47. Fabricated chassis rear view .....	58
Figure 48. Fabricated chassis front view .....	58
Figure 49. Chassis database .....	59
Figure 50. Import window setting configuration .....	60
Figure 51. Imported cad of the chassis in ADAMS/ View .....	61
Figure 52. Merge window – suspension merged .....	61
Figure 53. Fully merged model.....	62
Figure 54. Importing of flexible compatible chassis cad .....	63
Figure 55. Flex chassis part quantity .....	64
Figure 56. Single highlighted solid .....	64
Figure 57. Single solid under Chassis part.....	65
Figure 58. ViewFlex window configurations used.....	65
Figure 59. Full flexible model completed .....	66
Figure 60. Course used and derived from Race Technology’s software .....	67
Figure 61. Sector times per driver.....	68
Figure 62. Equations of maneuver for lane change .....	69
Figure 63. Lateral input for slalom maneuver.....	69
Figure 64. Lane Change input for rigid model.....	70
Figure 65. Lane Change input for flexible model.....	70
Figure 66. Slalom input for rigid model.....	71
Figure 67. Slalom input for flexible model.....	71
Figure 68. Slalom lateral input for rigid model.....	72
Figure 69. Slalom lateral input for flexible model.....	72
Figure 70. Lane change load transfer results for rigid model .....	73
Figure 71. Lane change load transfer results for flexible model .....	73
Figure 72. Slalom load transfer results for rigid model .....	75
Figure 73. Slalom load transfer results for flexible model .....	75

## LIST OF TABLES

Table 1. Suspension component names.....	21
Table 2. Idealized joints .....	23
Table 3. Primitive joints.....	23
Table 4. Equilibrium results per corner.....	57
Table 5. Lane change Load transfer values per corner .....	74
Table 6. Slalom load transfer values per corner .....	76

# **1. INTRODUCTION**

---

## **1.1 Background**

Racing is as old as the internal combustion engine. As the years went by, vehicles got faster, technology got better, and racing classes began to develop. The most prominent are Formula 1, Indy Series, NASCAR and at the academic level the College design Series Formula SAE competition (6). The objective of each is simple – produce the fastest race time within the rules of each’s specific competition rules.

With the development of tools such as finite element analysis, physic-based models, and computer aided design, the cost of design has been reduced significantly. Finite element analysis has allowed improvement to the torsional rigidity of the chassis whereas physic-based system level models have allowed component sizing to optimize weight and/or weight distribution. With the fast growth of these tools, their respective analysis capabilities have grown in complexity as well; therefore, requiring race car engineers to keep up with their development and utilize their complete capability

## **1.2 Problem Statement**

The purpose of this thesis is to develop a methodology that can be replicated and built upon for the design of a racecar which will minimize race times. One approach is to minimize weight by creating a chassis which maintains handling while minimizing chassis torsional stiffness. An approach to accomplishing this is to create a rigid chassis model that can be easily modified to a flexible multi-body dynamic model. The methodology developed here will provide a quick and efficient process for creating a

multi-body model that only requires the needs of two subsystems, the suspension and chassis, for full vehicle analysis. This will lay the foundation for development of more complex full vehicle models. These models can then provide a less resource extensive tool for the design of a racecar, from conception to validation.

Due to the proprietary nature and published racecar analyses using multi-body dynamic models, design methods are not available to most racecar engineers. Vast amounts of literature are available on multi-body analyses, but are generally only available for a specific design, thus do not provide a general methodology available to designers. The methodology developed and demonstrated here will provide a tool to other racecar engineers and their respective communities for open review and improvement to refine a process for a full vehicle analysis

### 1.2.1 Limitations and Defining Measurement of Performance

Due to the available resources, certain limitations were identified as to frame the boundaries of this paper. They are listed in order of impact from highest to lowest.

- Availability of computing resources
- Availability of software
- Availability of rigid chassis race car data

While defining “handling” in this paper is important for demonstrating the methodology, the approach developed is independent of the definition. For example, handling could refer to a vehicle’s oversteer/ understeer characteristics. However, for this paper, it will be defined as the measure of how well a system performs based on the transient and steady state load transfer while cornering. Ideally, load transfer should be

minimized as much as possible, to maximize lateral load carrying capability (cornering g's) while controlling under/oversteer.

### 1.3 Literature

A study done by Pablo Luque et al (3) considers the design of a vehicle participating in the 'Copa de Espana de Montana para vehiculos CM' with a specific set of vehicle parameters. This study shows the entire process for the optimization of a design in relation to total weight & distribution, stiffness, and strength. They show the design proceeding from CAD to FEA to a final mechanical analysis system while showing where to implement optimization in the process. Their final step was the actual manufacturing and testing of a prototype to verify their proposed methodology. Another study by Jose Lucas Lima Berretta e Guilherme Canuto da Silva (1), shows how a suspension model is developed in a virtual environment (ADAMS/Car). This paper shows how a model is made using ADAMS/Car. However, the simulation is not presented making evaluation challenging. Another paper by Mohammad Al Bukhari Marzuki et al (4), shows an analysis to find the mode shapes of the chassis using FEA (ANSYS). The paper explains the details very well and how to use the Fast Fourier Transform (FFT) with frequency response.

All papers discussed have some things in common, that they use multiple types of programs and require extensive knowledge of each, but leave out the information needed to replicate their results. What has yet to be done is a method in which the data parameters related to only the suspension and chassis subsystems, can be used into a high-fidelity level system that can produce results that can be applied in an iterative design efficiently and effectively. The system should be able to take

advantage of as many parameters as possible required to understand the behavior and feasibility of design. Like any design, they are only useful until a prototype is created and tested. The methodology should then be able to be refined to account for the test data. What is proposed is a new methodology that will not only reduce the design process, but design at the system level while data available at the detail level, all with the method transparent and reproducible.

## 1.4 Model Process Development

### 1.4.1 Software Selection

The software available to use were subprograms from the main software package, Automated Dynamic Analysis of Mechanical Systems, or better known as ADAMS from MSC Software. Of the subprograms available, two were considered. The first, ADAMS/ View, is a great tool for those whose requirement is to perform analysis on mechanical systems with a simple graphic user interface. Most new modelers start off in ADAMS/ View to learn and understand its capabilities. ADAMS/ View is limited like other programs by the modeler's ability, the solver, and available processing power. ADAMS/ View is available to students at no cost and to professionals at a cost commensurate with its capabilities.

The other subprogram that is widely used for vehicle type analysis is ADAMS/ Car. ADAMS/ Car is for solving full or partial vehicle system analysis including the complexity of tires. As such, the software requires information on all subsystems of a vehicle to perform a simulation, thus increasing model complexity. ADAMS/ Car also requires the modeler to have an extensive understanding of the software.

As an example of the level of detail ADAMS/Car requires, Figure 1 shows a window of information pertaining just to the powertrain parameters.

	real value	remarks
<b>pvs clutch capacity</b>	1.0E+005	(none)
pvs clutch close	0.25	(none)
pvs clutch damping	1000.0	(none)
pvs clutch open	0.75	(none)
pvs clutch stiffness	1.0E+004	(none)
pvs clutch tau	5.0E-002	(none)
pvs ems gain	5.0E-003	(none)
pvs ems max throttle	100.0	(none)
pvs ems throttle off	1.0	(none)
pvs engine idle speed	1200.0	(none)
pvs engine inertia	1000.0	(none)
pvs engine rev limit	1.35E+004	(none)
pvs final drive	4.25	(none)
pvs gear 1	2.75	(none)
pvs gear 2	1.95	(none)
pvs gear 3	1.5	(none)
pvs gear 4	1.28	(none)
pvs gear 5	1.142	(none)
pvs gear 6	1.061	(none)
pvs gear r	-3.0	(none)
pvs max throttle	100.0	(none)

Display: Single and  Left  Right  Both

Value Type:  Real  Integer  String  All Data Types

Name Filter: \*

Figure 1. ADAMS/ Car powertrain parameters

From the new user perspective, this can prove daunting as the information shown may not readily be available or is proprietary.

ADAMS/View was decided to be the best option versus ADAMS/Car for this thesis due to the following:

- Level of Complexity
- Availability
- Subsystem Focus

#### **1.4.1.1 Level of Complexity**

This is more of an umbrella term in that it encompasses the ease of use as well as how complex a model can be. Because ADAMS/Car looks at the full vehicle, it inherently increases model complexity with difficult software usage. Difficult is in the sense that it requires access to a high-level computer, introduces model complexity and a user interface that is not friendly to the new user. Also, ADAMS/ Car utilizes other sub-programs that the user will have to become familiar with such as ADAMS/ Driveline.

#### **1.4.1.2 Availability**

Due to ADAMS/ View being available as an entry level tool it was obvious to use the software that would be more readily available., ADAMS/ View is interfaced in a way that allows the user to create what they need easily to perform a multi-body dynamic analysis.

#### **1.4.1.3 Subsystem Focus**

ADAMS/Car, requires information from all subsystems whereas the methodology presented in this thesis, only requires the chassis and suspension subsystem. ADAMS/ View allows the user to create only what is needed to perform their analyses. In this case, the user does not need to make assumptions on other subsystems in detail but can use system level parameters, such as weight, in their model.

Even though ADAMS/ View was selected as the preferred software for the methodology presented, it must be acknowledged that ADAMS/ Car has superior



capabilities. The methodology presented here clearly establishes the ground work to extend to ADAMS/Car.

#### 1.4.2 Development

The development of this came about through years of experience within the University of New Mexico's FSAE program. From 2014 to the upcoming 2020 FSAE vehicle, ADAMS/View and ADAMS/Car has been utilized on the designs of seven cars. Although there is much history of the usage, there has not been a process developed in which, as mentioned in the literature, a method where the chassis and the suspension are co-simulated and analyzed efficiently. Due to poor documentation of the usage of the ADAMS software; this thesis aims to fully set a methodology and demonstration for anyone to recreate, while remaining open to review and improvement.

Originally the methodology only used the suspension parameters. Over time, the process for the suspension became faster due to the need for more design iterations. The inclusion of the chassis has made it possible for even greater representation of simulations performed for a design; therefore, letting designers not only use numerical data, but visual verifications of behavior as well. This is achieved with the simulations animating the model based on design parameters.

### 1.5 Demonstration of Process

#### 1.5.1 Reference used

The methodology will be demonstrated using the University of New Mexico's 2018 FSAE vehicle for the following reasons and corresponding explanation:

<ul style="list-style-type: none"> <li>• The modeler has no knowledge of design decisions.</li> </ul>	<p>This is to show that one does not need to know a substantial amount of information regarding the actual reference used to properly use the proposed modeling process.</p>
<ul style="list-style-type: none"> <li>• There is an existing prototype of the 2018 FSAE car.</li> </ul>	<p>Though no experimental verification is done, the existing prototype helps aid in verifying the models.</p>
<ul style="list-style-type: none"> <li>• All data pertaining to the design of the vehicle is available as well as the designers for further contact (if need be).</li> </ul>	<p>It's important to have all forms of available data to help in the development of the model.</p>
<ul style="list-style-type: none"> <li>• The 2018 FSAE car has different front and rear suspension configuration.</li> </ul>	<p>This helps show the variations a suspension design can have and provide an example of how to model them.</p>

All the reasons specified above contribute to the fact that considering this was the first application of the method, it was important to have a reference that had access to all possible sources of information to address potential problems that may arise further refining the process.

### 1.5.2 Demonstration Value to Development

Since this methodology has never been used, demonstrating its ability as well as contributing to the development of the method proved necessary. Test running the method showed possible errors and flaws that were able to be addressed and refined. The demonstration also showed the possibility of future improvements. This thesis discusses these other areas of interest but were left for further study. Ultimately, the demonstration shown provides an example to the designer how the method is applied, can be replicated, and the typical results available from it.

## **2. MODELING PROCESS**

---

The process described in this chapter is under the assumption that the modeler has an intermediate understanding of the ADAMS/ View software. The flow path shown in Figure 2 will be referenced and followed while also broken down in detail.

As a quick overview, the diagram begins with the development of the suspension. The suspension process covers everything needed; from gathering all information needed to model checks. The chassis is then created by a different method described in its portion. Both the suspension and chassis will be separately created as to avoid complications and difficulty but will later be merged into one model. The merged model can be simulated by using equations that define the maneuver the model will perform. Once the first model has completed the diagram's work flow, the second model (the flexible model) can repeat the same process but with some extra steps to account for the chassis as flexible. Once the flexible model has completed the results section, the first model, rigid, can be compared to the second, flexible.

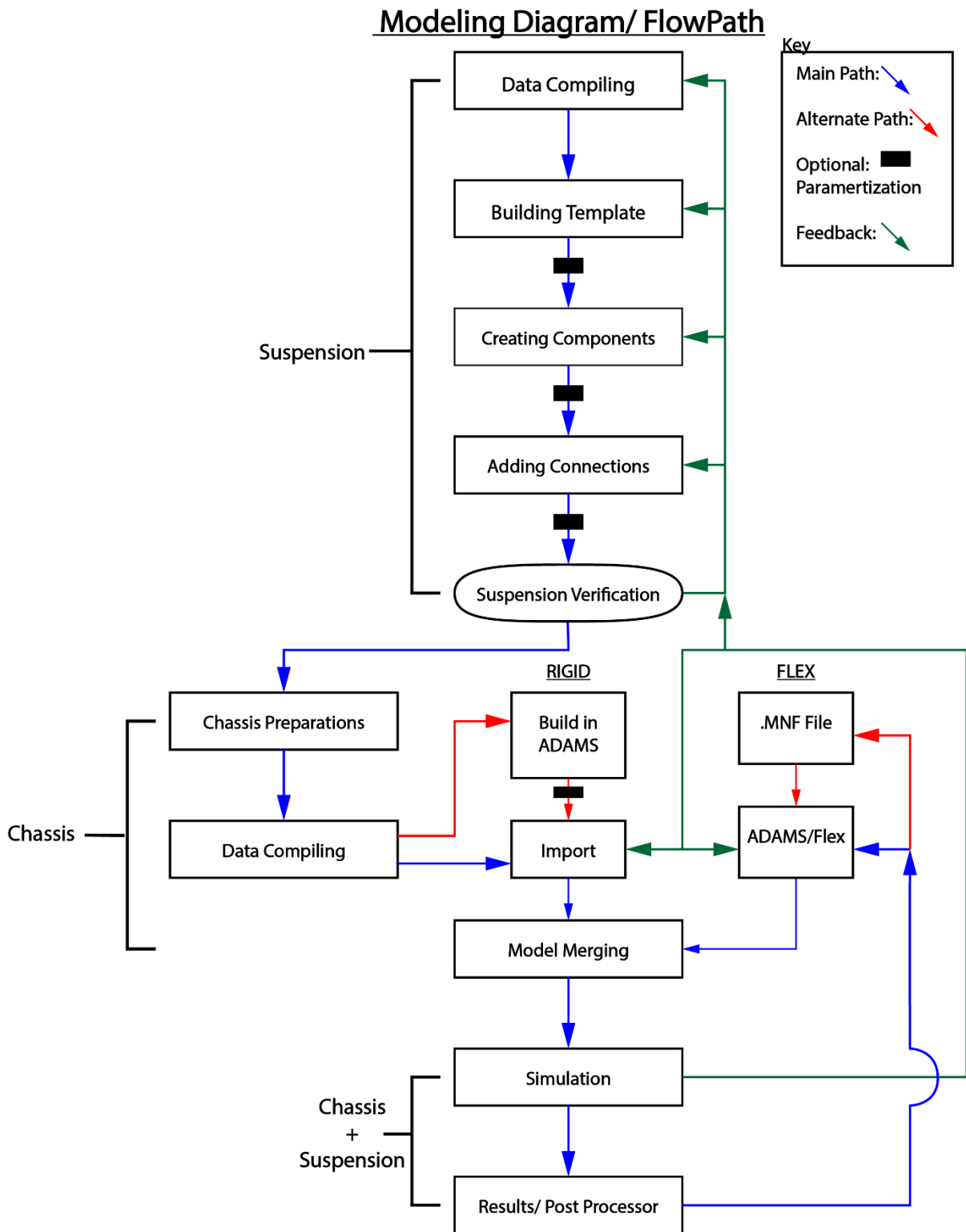


Figure 2. Methodology work flow diagram.

The diagram can be broken down into three sections (Suspension, Chassis, & Suspension + Chassis), followed by their respective sub-sections. Each sub-section comprises a quick overview of what is covered followed by a semi-detailed guide. The semi-detailed guide only covers the main path shown in the diagram.

As mentioned earlier, the process presented here is what was determined to be the most efficient way to build the model. The process shown is open for criticism and improvement, so the reader should not take it at face value but is encouraged to explore and replicate the results. Due to the nature of modeling and the techniques available, the diagram shown is to be used only as a general work flow as special cases are not covered by it. A pre-existing FSAE prototype vehicle was chosen that covers multiple special cases so as to familiarize the designer with the process. Details will be covered in the demonstration/ application portion.

## 2.1 Suspension

This section covers how the suspension model is created step by step. Each sub-section under this section will pertain only to the suspension and its accompanying parts. Due to the complexity of the suspension, it's especially important to take special care when creating the model as errors can cause the model to not function correctly or not at all.

### 2.1.1 Data Compiling

*This sub-section considers compiling the information required to create the suspension. Ideally, all the information needed to create the suspension should be readily available to build the model efficiently and effectively. Data parameters that*

*are not known exactly should be estimated as best as possible. Not having this data will cause delays in the process that could potentially lead to erroneous results. The type of information needed will be introduced as well as its necessity.*

The information to be collected and what they entitle is as follows:

#### **2.1.1.1 System's Requirement Documents (SRD)**

This document or any similar/ equal level document should contain information that states what the design/ system is intended to achieve. As the name implies, it should be capabilities set at the system level. Example of system level parameters are vehicle weight, driver weight, lateral and longitudinal capabilities, center of gravity location, etc. This document serves to establish the type of simulations that must be performed to meet the required design goals.

#### **2.1.1.2 Suspension Parameters**

Parameters needed:

- Suspension points – spatial points that dictate the geometry of the suspension.
- Weight Distribution – ratio of weight distributed between the front and rear relative to the center of gravity (CG)
- Track Width & Wheelbase – width and length of the vehicle respectively.
- Spring Rates – stiffness values determined for the springs well as the anti-roll bar.

- Damping Rates – force versus velocity curves for dampers or damping coefficients
- Joints – connections between suspension components that dictate the degrees of freedom each component has relative to each other. Ex. spherical bearings and cylindrical bearings.

### **2.1.1.3 Reference Point**

For the coordinate frame of reference, it should be noted that Z = vertical, Y = lateral, and X = longitudinal. Also, the center of gravity (CG) of the vehicle should be set at (0.0, 0.0, #. #) where ( X, Y, Z) is the order of points set. Only the Z, the vertical, should change when setting up the CG. Once the CG point is created, all points are then determined from the CG's location. This is used as suspension software only may only output the geometry of either the front or rear as standalone and not consider the spatial points from a full vehicle setup. Figure 3 shows this frame - but note that the modeler can create any reference frame that works best for them. The reference frame presented here was determined to be the best for the methodology presented.

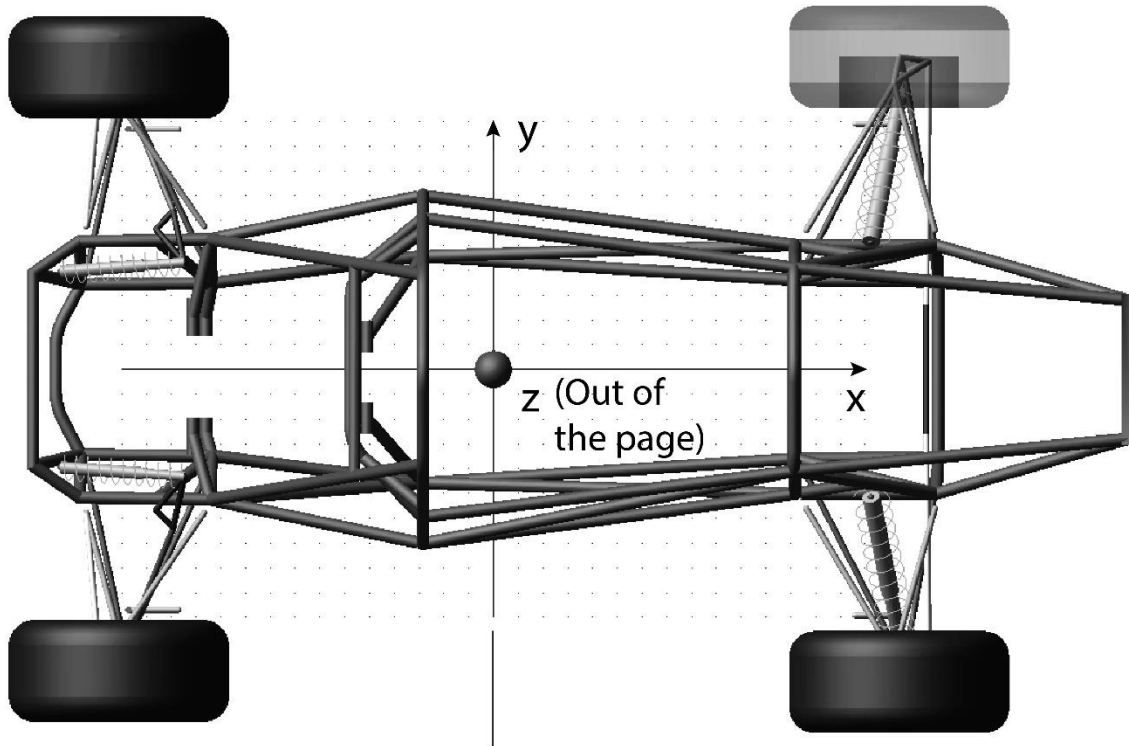


Figure 3. Reference frame

### 2.1.2 Building the Template

*This sub-section sets up the template to begin creating the suspension model. The template will essentially act as the backbone to the entire model. Optional: Parameterization and how the model is parameterized in the template allows changes to be made with ease.*

To create the template, the following information is needed:

- Reference Frame
- Suspension Geometry
- Weight distribution.
- Wheelbase and track width



Keeping the reference frame in mind, determine if the suspension geometry is correct.

For the geometry to be correct, the following criteria needs to be met:

- X coordinates for front to rear wheel center distance must equal the wheelbase
- Y coordinates for the left to right wheel center distance are the same for track width (can only pertain to either front/rear suspension or full suspension if applicable).
- Geometry is fit to the reference frame.

To calculate how the suspension geometry needs to fit to the reference frame, the following should be reviewed.

*Calibrated Front Points<sub>x</sub>*

$$= (\text{Front weight ratio}) * (\text{Wheelbase}) + (\text{Front suspension geometry}_{x})$$

*Calibrated Rear Points<sub>x</sub>*

$$= (\text{Rear weight ratio}) * (\text{Wheelbase}) + (\text{Rear suspension geometry}_{x})$$

These equations fit the suspension points into the reference frame as all points. Only the x-coordinates of the points should be moved as it is assumed the y-coordinates are accurate to the track width set. If the y-coordinates do not meet the track width criteria, adjust accordingly or a review of the suspension points may be needed.

Note that the front suspension should have positive X-values and the rear negative X-values due to the reference frame. Finally, choose either the left or right side of the suspension to be created, do not do both. Under the assumption of symmetry of the XZ plane, the other side is simply mirrored. At this point, the front suspension geometry should have all positive X-value coordinates while the rear suspension has

all negative x-value coordinates. The distance from the front wheel center to the rear wheel center should be the same as the wheelbase. One side of the Y-coordinate of the suspension should be half of the track width. With the geometry now aligned with the reference frame, creating the template is next.

To create the template, the markers feature shown in Figure 4 will be used to create the spatial point for the suspension. With the markers set to “attach to ground” begin placing point arbitrarily in space.

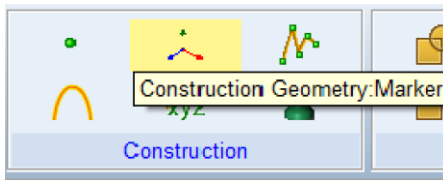


Figure 4. Marker Icon

The exact coordinates do not need to be used at this moment. Along with placing the markers, it is important to start a naming convention that will be applied to the template. Any naming convention can be used at the discretion of the modeler, but the convention used here is shown below:

*Y(type)XX(section)\_subsystem(optional)\_Component Name*

Where “type” refers to what kind of feature (m = marker, j = joint) while “section” refers to one of the four corners of the vehicle, shown below:

Front Left (FL)

Front Right (FR)

Rear Left (RL)

Rear Right (RR)

This will make attaching bodies and joints much easier later in the process. Once the set number of points and respective names are made, using the table editor in Figure 5, modify the points of the suspension geometry to the exact coordinates.

	Adams Id	Loc X	Loc Y	Loc Z	
EXAMPLE_MARKER_1	1	-150.0	150.0	0.0	
EXAMPLE_MARKER_2	2	200.0	200.0	0.0	
EXAMPLE_MARKER_3	3	-150.0	-50.0	0.0	
EXAMPLE_MARKER_4	4	250.0	-50.0	0.0	

Figure 5. Table editor

The table editor makes it easy to input and verify the coordinates. Once one side of the front and rear suspension markers (either left or right) have been placed, the other side is added.

Since the left and right are symmetrical about the XZ plane, there are a number of ways to proceed from here. One is simply copying and pasting the existing points and modifying the Y coordinate while also changing the respective names of the points. Another is parameterizing points to mirror its respective point. Both have pros and cons. For the first option the pros are easy and simple but suffer the con of difficulty of future modifications. The second option has the con of a substantially longer creating process, but the pro is the ease of future modifications. The modeler will have

to decide which is best suited for their need, but the more difficult option will be discussed as the simpler method should be straight forward.

To start, the same number of markers for one side is placed again on the working grid (recall position does not matter yet). Therefore, if the left side of the total suspension is 50 markers, then the right side should have 50 markers placed anywhere on the working grid. Repeating the same process of renaming the points but this time indicating that these new points are the other side of the suspension. With all points named, next parameterize the decided side to the other. To do this, manually input the function into the specified marker location, shown below:

LOC\_MIRROR({ARRAY}, OBJECT, STRING)

Where:

ARRAY = The component to be parameterized to. Denoted as "item".location.

OBJECT = The reference component that defines how the item is parameterized about.

STRING = plane mirrored about in respect to the Object orientation, denoted using either "XY", "YZ", "XZ".

Using the assist feature in the Expression Builder, a window appears where the modeler can input the parameters that will automatically create and populate the function above, shown in Figure 6.

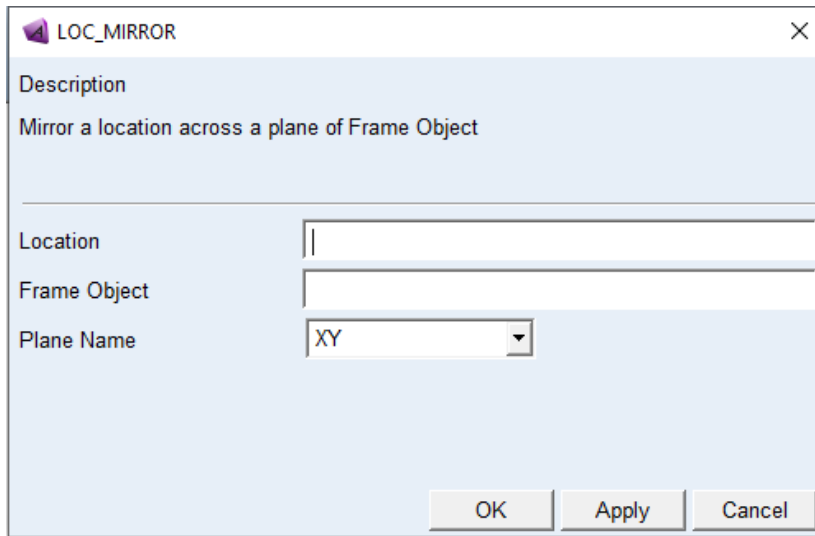


Figure 6. Parameters

Once all points for one side are parameterized to their respective point, check that all points are where they need to be. Additionally, the CG marker should be placed as mentioned in sub-section 2.1.1.3. Make sure all points are correctly located as most major errors occur from the geometry incorrectly or even slightly misplaced. The Template should be done and ready for the next phase.

### 2.1.3 Creating the Parts

*The focus in this sub-section is creating the parts for the suspension while keeping in mind the mass properties of the parts. Mass can have big effects to the system and if any masses are incorrect, errors such as force imbalance can occur or cause incorrect results. Parts size affect the inertia values as they are pre-determined from them. The inertia values can be altered to reflect more complicated parts in a simple looking shape. Finally, the appearance such as color of a part is only for visual purposes but do help in visual verification and identification.*

To begin, create the center of gravity of the model with a spherical body attached to the corresponding template marker. The size of the solid does not matter as the mass and inertia values will be modified but the size kept within reason to allow for easy access and viewing. The CG's mass will need to be modified to include the full weight goal as well as the driver's weight minus the suspension weight. The suspension weight is subtracted as this weight will be represented by the modeled components of the suspension as well as the tires/ wheels. This is shown below:

*Vehicle Weight*

$$= (\text{Design Goal Weight}) + (\text{Driver Weight}) - (\text{Suspension Weight}) \\ - (\text{Tire Weight})$$

With the CG created, the process is the same for each corner of the model therefore only one corner will be mentioned. It is up to the modeler to repeat three more times unless otherwise needed. Again, apply a naming convention as soon as a new part is created. An example of a naming convention is shown below:

*XX(section)\_Component Name*

Component naming is up to the discretion of the modeler, but Table 1 shows the component names used here as they are also compatible with ADAMS/ Car for future work with.

Table 1. Suspension component names

Component Group	Component	Adams/View
Upright	Lower Ball Joint	lca_outer
	Upper Ball Joint	uca_outer
	Tie Rod	tierod_outer
Bell Crank	BC to Droplink	ARB_droplink
	Spring to BC	damper_outboard
	BC Pivot	BC_center
	P-Rod to BC	prod_inboard
UCA	Forward	uca_front
	Rear	uca_rear
	Outer	uca_outer
LCA	Forward	lca_front
	Rear	lca_rear
	Outer	lca_outer
P-Rod	Outer	prod_outboard
	Inner	prod_inboard
Tie-rod	Outer	tierod_outer
	Inner	tierod_inner
Tire	Center	wheel_center
ARB	droplink	ARB_droplink
	blade	ARB_leaf_link
	center	ARB_center
	bar	ARB_bend

No specific order is required in creating the components, but it is crucial that the right bodies are added correctly. As an example, when creating the control arms, there will be two cylinders that will need to be one part. Therefore, take special care to identify whether a new part needs to be added or to an existing part. The only bodies that should be used for creating the parts is the Cylinder and Sphere bodies shown in Figure 7.

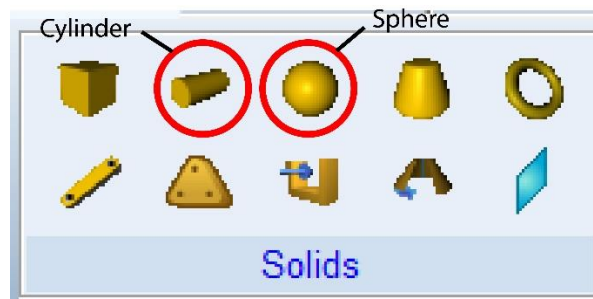


Figure 7. Solids

Using the template made in the previous section, it is easy selecting the points for the solid parts. Again, sizing does not matter as the mass and inertia values can be modified but if the size of the parts are known, then the parts can be sized in ADAMS.

Once all parts are created, verify that all parts have the correct mass and inertia values. Before moving on, one last part needs to be created and that is the ground. The solid that will act as the road should be added to the ground with as little a gap between the tires and the ground vertically. A large gap may cause errors when simulating.

#### 2.1.4 Adding Connections

*This portion of the process covers creating the constraints for the parts created in the previous section. Defining how the parts are connected to one another through joints that dictate the degrees of freedom. Also, the addition of Forces will also be covered. Forces refers to the shocks, anti-roll bar, contacts, applied forces (input), and any other force related features under the Force tab.*

##### **2.1.4.1 Joints**

When creating the joints, it is crucial to understand how the parts are interconnected - specifically, identifying what degrees of freedom are allowed between parts. ADAMS has available idealized joints but if a more specific constraint is required, it also has access to primitive joints. Table 2 and 3 from the ADAMS help menu should help determine what degrees of freedom are removed, by type.



Table 2. Idealized joints

**DOF Removed by Idealized Joints**

		Rotational DOF Removed			
		0	1	2	3
Translational DOF Removed	0	(Part)			
	1			Planar	
	2			Cylindrical	Translational
	3	Spherical	Hooke/Universal Constant Velocity	Revolute	Fixed

Table 3. Primitive joints

**DOF Removed by Joint Primitives**

		Rotational DOF Removed			
		0	1	2	3
Translational DOF Removed	0	(Part)	Perpendicular	Parallel Axes	Orientation
	1	In Plane	In Plane + Perpendicular	In Plane + Parallel Axes	In Plane + Orientation
	2	In Line	In Line + Perpendicular	In Line + Parallel Axes	In Line + Orientation
	3	(Spherical)	(Spherical) + Perpendicular	(Spherical) + Parallel Axes	(Spherical) + Orientation

Once the joints have been determined, apply them between the necessary components until all parts are constrained properly.

#### 2.1.4.2 Flexible Connections

The features to use from this tab is the translational and rotational spring-damper. The translational spring-damper, here on known as the shocks, will be used to connect the respective suspension parts to the chassis. The modeler can create the shocks first and afterwards modify the stiffness and damping values to the actual values. The rotational spring-damper, or the anti-roll bar, should also connect to the respective

suspension components. Special attention should be taken when defining the vector that the rotational spring-damper will rotate about. Again, values can be modified afterwards.

### 2.1.4.3 Special Forces

The special forces to be used are the contacts between parts, shown in Figure 8. In this case, the contact between the tires and the ground.

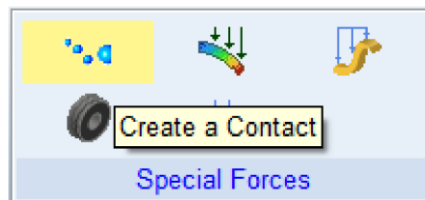


Figure 8. Contact feature

Once contact between all four tires are made to the ground, certain parameters need to be adjusted. Recall that one of the limitations of this model is the lack of a tire model; therefore, a standardized value for the stiffness and damping of a tire is used, the values used are shown in Figure 9.

Stiffness	(179904(Newton/meter))
Force Exponent	2.2
Damping	(250(Newton-sec/meter))
Penetration Depth	3.937007874E-003

Figure 9. Tire value

The force exponent and penetration depth are left default but can be used if the modeler chooses to. Rename the created contacts by the respective corners that they were created for.

#### 2.1.4.4 Applied Forces

A force vector is used as the system input for the model. This feature is shown in Figure 10, which should be created and attached to the CG of the vehicle with the ground as the reactive component. The orientation chosen for the force vector should be the same as the reference frame for the model.

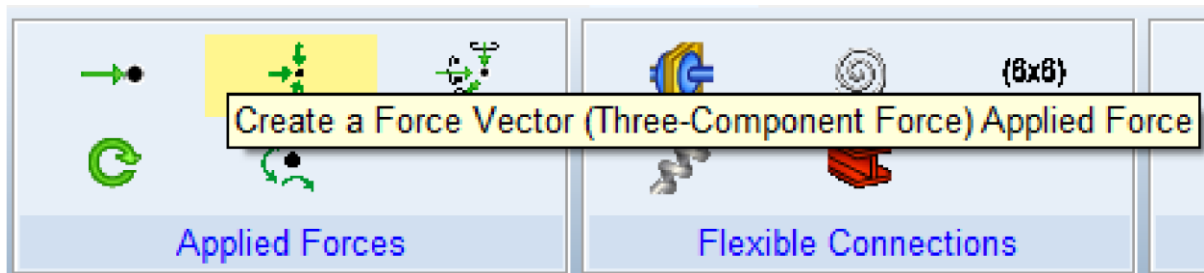


Figure 10. Force vector feature - model input

#### 2.1.5 Suspension verification

*Due to the complex geometry the suspension can have, verification of the newly created suspension model will be covered here. Basic tests that check suspension parameters as well as suspension behavior. Some tests involve checking values while others use visual verification with data to backup if the suspension model is ready to go!*

##### 2.1.5.1 Simple Simulation Test

A straight forward test is simulating the model. To perform this test, have a large number of steps that allow the individual parts to interact at a speed that can be observed. This test checks to see if any parts are not connected properly and will either show parts partially connected or not at all further causing parts to fall in space and through the ground. Identify any of these errors and address them appropriately to the respective sub-sections.

### 2.1.5.2 Equilibrium/ Load distribution Test

Another test is the static equilibrium test. To initialize this test, Figure 11 shows how the simulation control window is configured. Once the simulation is successfully completed, move to the post-processor and plot all contacts for the tires in the Z direction. Since no maneuvers have been defined yet, the contact loads should show how the weight of the model is distributed. Verify the loads are correct with the Front/Rear weight distribution ratio with this plot.

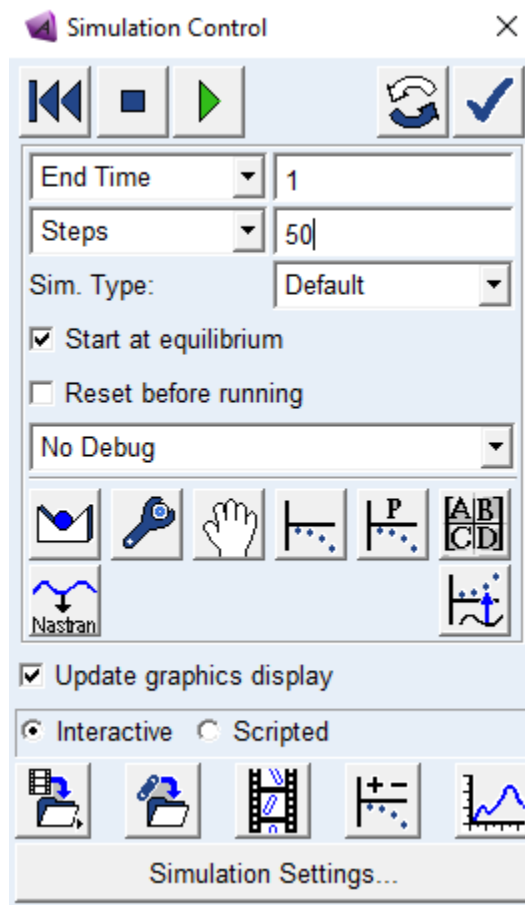


Figure 11. Simulation control window – equilibrium configuration

This test checks to see if the geometry of the suspension is correct or modifications to the solver are required.

From experience, it has been shown that cases where the load distribution was significantly off, can be attributed to suspension points placed incorrectly in space. For errors that have significant differences between left and right sides is most likely to be related to incorrect spring rates set for the shocks. Finally, the worst error that can be observed is the output from the ADAMS/Solver, notifying the user that the simulation has failed due to certain parts experiencing forces that are extremely high or that the solver could not determine the equilibrium from the set amount of iteration attempts. Regarding the high force components, again it may be attributed to suspension points misplaced but may require further diagnosing. The iteration attempt can be directly addressed by modifying the “Maxit” value, increasing the number of attempts the solver will try to find the equilibrium as shown in Figure 12 as “1”. Another option is the “Error” Order of magnitude shown in Figure 12 as “2”, which will essentially loosen the tolerance the solver will try to find the equilibrium within.

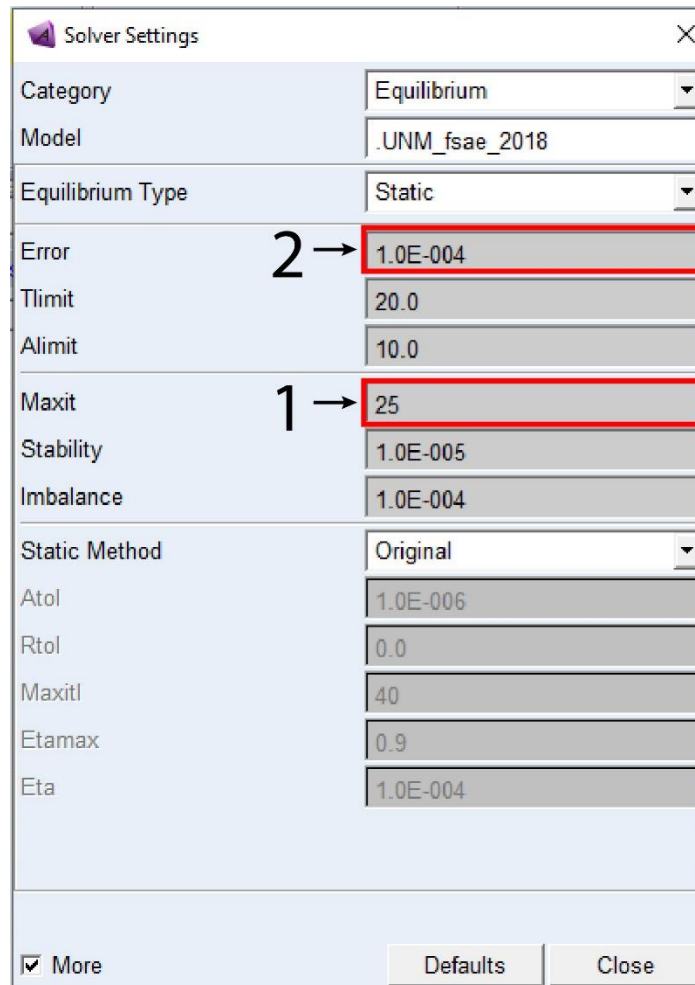


Figure 12. Solver setting adjustments

## 2.2 Chassis

The chassis portion of the methodology is rather open ended. The methodology presented here was determined to be efficient for later combining the chassis and suspension models. The technique requires the use of CAD software of the user's choosing. Other alternatives are available. One alternative to CAD software is using the chassis points that dictate where in space members are connected to. Using the points, the chassis can be created in ADAMS/ View via the Bodies tab while also making sure, members are added as "Add to Part". Note that the type of chassis that will be described will be a space frame chassis. Monocoque chassis would require the

use of CAD Software due to the complex geometry they inherit which is not possible in ADAMS/View without getting creative.

**IMPORTANT:** The process described here must be repeated twice -first for the rigid chassis, and a second time for the flexible chassis. Once the rigid version has been completed the results section or sub-section 2.3.2, then the flexible chassis can be created starting back in sub-section 2.2.3. This is seen on the main path in Figure 2. Differentiation between the two types will be identified.

### 2.2.1 Chassis Preparations

*To build the chassis, certain preparations need to be done. The preparations will individually focus on the chassis itself. Any modifications or fixes to the chassis can be done in this portion.*

To prepare the chassis, it's best to create a new model within the database. Since the chassis will be imported from CAD, it's best to have a space where only the chassis will exist. This allows the modeler to visually see the chassis itself and make any modifications needed either in ADAMS/View or it's respective CAD software.

### 2.2.2 Data Compiling

The approach the modeler chooses determines the information required. Again, it's important to have all information ready to avoid any mistakes during the process.

#### **2.2.2.1 Chassis Parameters**

-Chassis points (Alternative) – These are points that dictate the structure of the chassis as well as how the chassis members are interconnected among the points.

-Chassis CAD – A cad model that is as close to the final product. File type will vary.

### 2.2.3 Import

*The option to import the chassis is one of the easiest ways of implementing the chassis. With the preparations complete, importing the chassis can be done. The rigid version will be covered here but there will be another sub-section for the flex version.*

#### 2.2.3.1 RIGID

To import the rigid chassis, the STEP AP214 file type will be needed. This file type allows the CAD model to be imported as one body instead of multiple parts. This makes it easy to constrain the chassis to the suspension when the time comes to merge them. It should also help cut down the need for computer processing. The feature that allows this to happen is the “Consolidate To Shells”. Figure 13 shows the window with the necessary configuration.

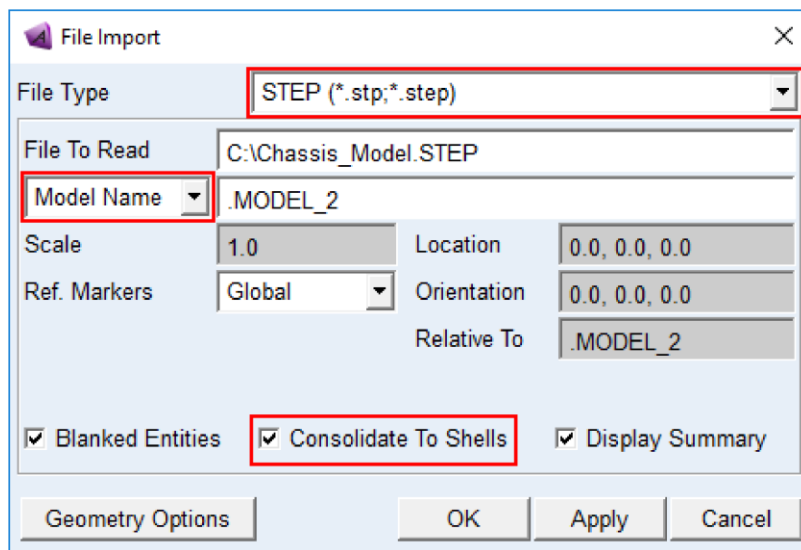


Figure 13. Import configuration



With “Model Name” selected, the new model created in sub-section 2.2.1 for the chassis should be in the window. Select OK and wait while the software imports the CAD model. Once imported, visual checks can be performed to make sure nothing was altered, or any errors occurred. Lastly, material properties are required to be defined per individual component. ADAMS/ View has a basic library of materials to choose from but if needed, the option of defining a material not listed is available, as shown in Figure 14. Once the newly imported model material has been defined, a center of mass marker is automatically calculated. The rigid chassis model is ready for merging.

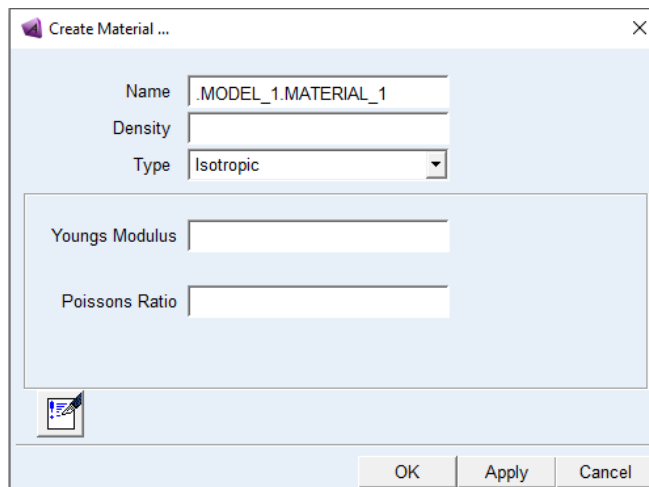


Figure 14. Material defining window

#### 2.2.4 Model Merging

*This portion describes merging the chassis and suspension models into one model in the data base. Note: Both the rigid and flexible require this step.*

To begin, the first thing should be to create a new model in the data base that both the suspension and chassis can be merged to. Therefore, MODEL\_3 shown in Figure 15 will act as the base model.

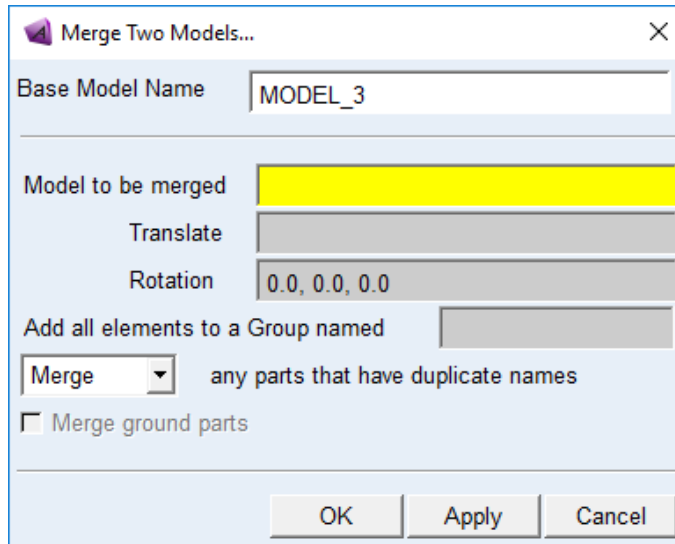


Figure 15. Merge window

Merging the suspension should be first as the chassis will be positioned relative to the suspension. It is done in this manner as it is easier to move one part than the number of parts the suspension entails. Once the suspension model is merged to the new model, do the same with the chassis model. The chassis may require repositioning when merged. Do so until it is properly located about the suspension. Once merged, the chassis will need to be merged with the existing chassis part in the suspension to transfer all constraints. Figure 16 shows the feature that does this.



Figure 16. Boolean, merge without contact

Lastly, verify the constraints for the chassis model are still applied appropriately from the existing suspension defined chassis part.

#### 2.2.4.1 FLEX special case

The body used as the CG in the suspension will need to be made into a new part. So once the flexible model is merged to the existing chassis part in the suspension model, the sphere body will have to then become a separate body entirely. This allows the software to mesh the body but will not if any bodies in the same part name are not merged properly or in contact.

#### 2.2.5 ADAMS/Flex

**IMPORTANT:** This sub-section should not be considered until the rigid chassis has completed up to the results sub-section. Any errors that occur in the rigid chassis will be substantially worse to deal with if there is a flexible body in the model. Address all problems in the rigid model and once all simulations are successfully completed, move to sub-section 2.2.5.1 to begin the flexible model process.

*This sub-section looks at converting the rigid chassis into a flexible chassis. ViewFlex is a subprogram that comes with the ADAMS Software package that allows incorporating components that act flexible without the need of a meshing/ FEA software.*

#### 2.2.5.1 Prep Work

The flex model version follows a similar path as the rigid but with more steps involved. Instead of importing the model using STEP AP214 file type, the Parasolid file type will be used instead. This file type will import the model as individual members of the chassis. Again, verify all members are where they need to be, and no anomalies are

within the newly imported model. Next is uniting all the members of the chassis into one part. Figure 17 depicts the feature that will be used for this step.

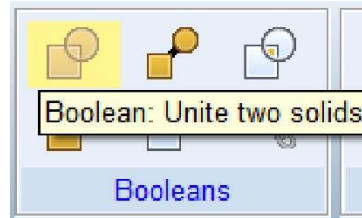


Figure 17. Boolean, merge in contact

The is important as ViewFlex, cannot mesh the file type used for the rigid model. A Parasolid file type allows the program to properly mesh and constrain the chassis when converted to flexible. Once all members are merged into one part, the part can have the material defined. The flexible chassis model is now ready for merging.

#### **2.2.5.2 ViewFlex**

ViewFlex is the subprogram within the software that allows certain parts to be converted from rigid to flexible.

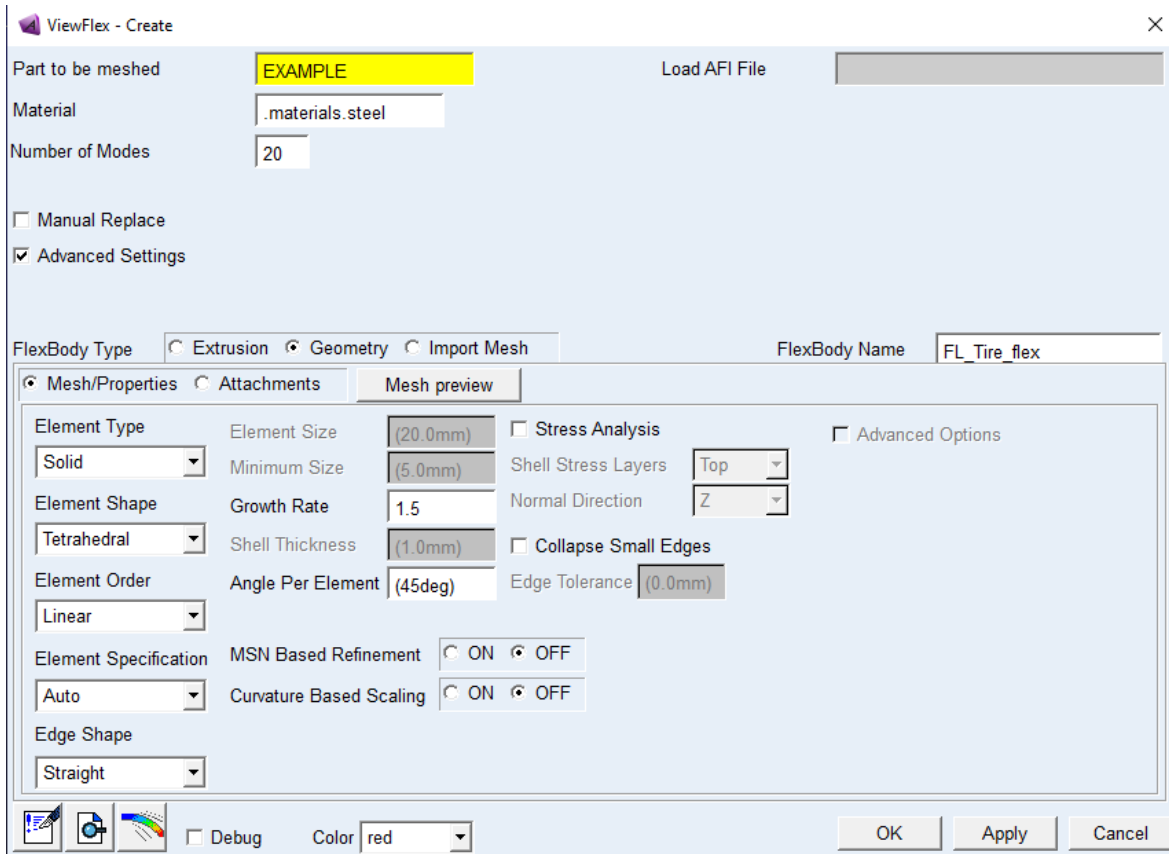


Figure 18. ViewFlex Mesh window

Figure 18 depicts the multiple areas where the program can be adjusted to properly mesh the wanted flexible part. Figure 19 depicts a cylinder that is meshed with each iteration modifying the element size until a mesh that is deemed acceptable and captures the curvature of the part.

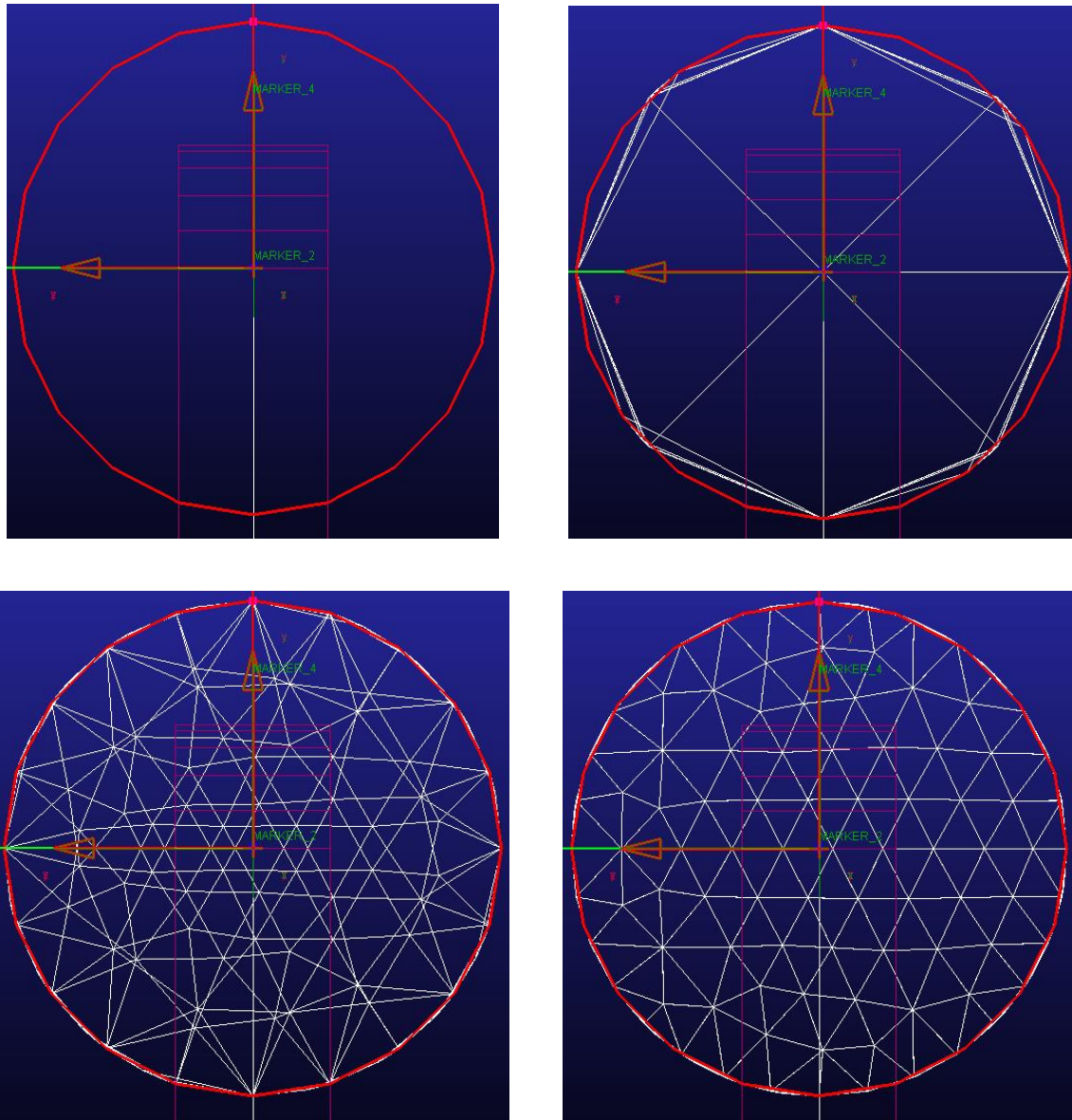


Figure 19. Mesh iterations

Once the mesh is created, next is the attachment portion that defines how the mesh is to interact with its surrounding parts. The connections for the model should be established that the modeler can use the “Find attachment” feature shown in Figure 20. This feature will auto populate the table based on the existing connections to the part that will be converted into flexible. Once the table is populated, hit apply and wait for the software to apply the changes.

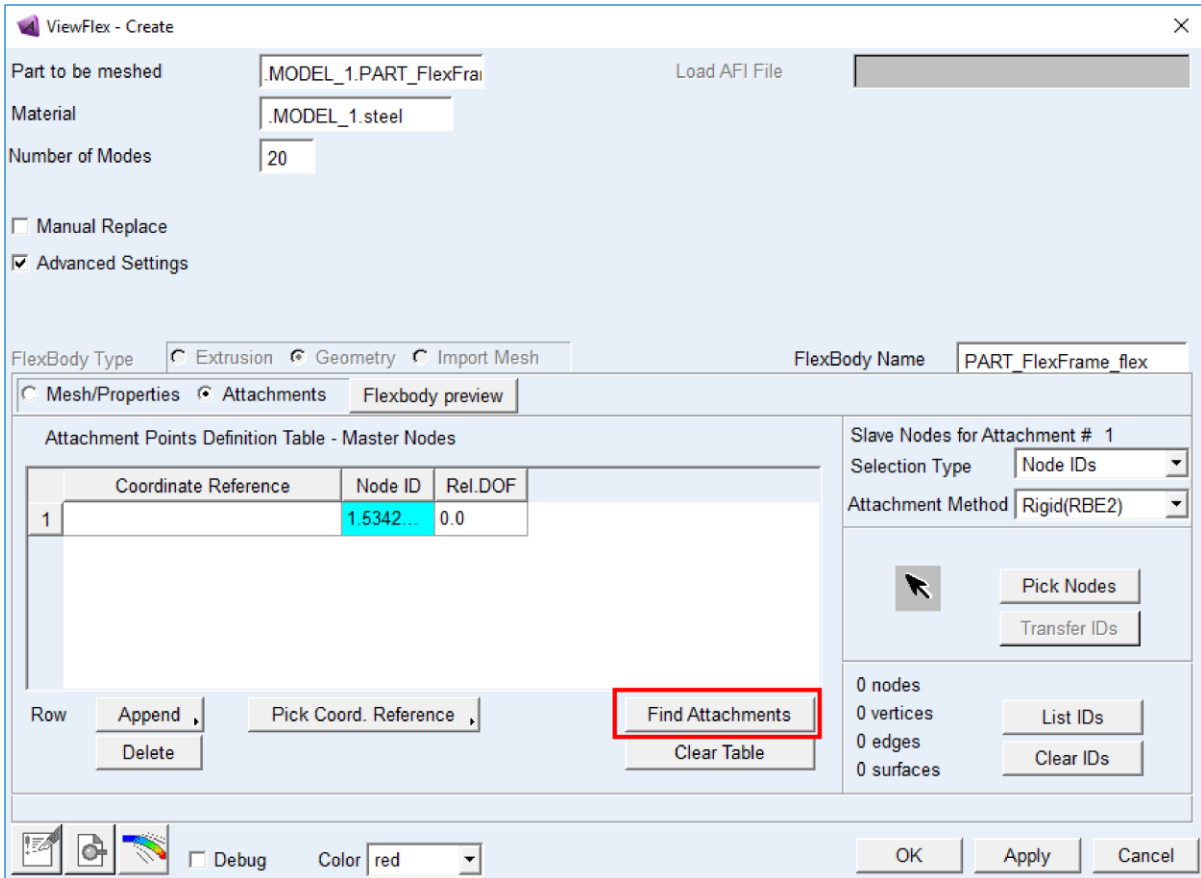


Figure 20. ViewFlex attachment window

When the software completes the conversion, the pre-existing rigid part will still be in the model tree but it will be deactivated and hidden as the new flexible body will take its place. If for some reason the modeler needs to make any edits or changes to the original part, it is still available to use.

### 2.3 Chassis & Suspension

The model(s) are now ready for simulation but must be defined before proceeding. In this simulation, all input is applied to the CG as noted from sub-section 2.1.4.4. For demonstration purposes two cases will be looked at: the Lane Change and the Slalom.

### 2.3.1 Simulating

Keep in mind when defining the simulations for the model(s), the frame of reference for the input. For the method of application, it's best to apply the equations from the behavior of a vehicle under such loads. Therefore, negative longitudinal acceleration is accelerating, where the rear tires of the vehicle will load more than the front tires, known as squat. Positive longitudinal acceleration is braking, where the front tires will load more than the rear tires, known as dive. For lateral acceleration, positive is a right-hand turn and negative is a left-hand turn.

#### 2.3.1.1 Lane Change

The lane change maneuver to be used is depicted in Figure 21. When defining the equations, the modeler needs to define two parameters to perform this study:

1. Number of G's in magnitude
2. Time the maneuvers are performed.

The second parameter is the key in defining a realistic lane change maneuver. This is explored further in chapter 3 but the information presented here is just conceptual.

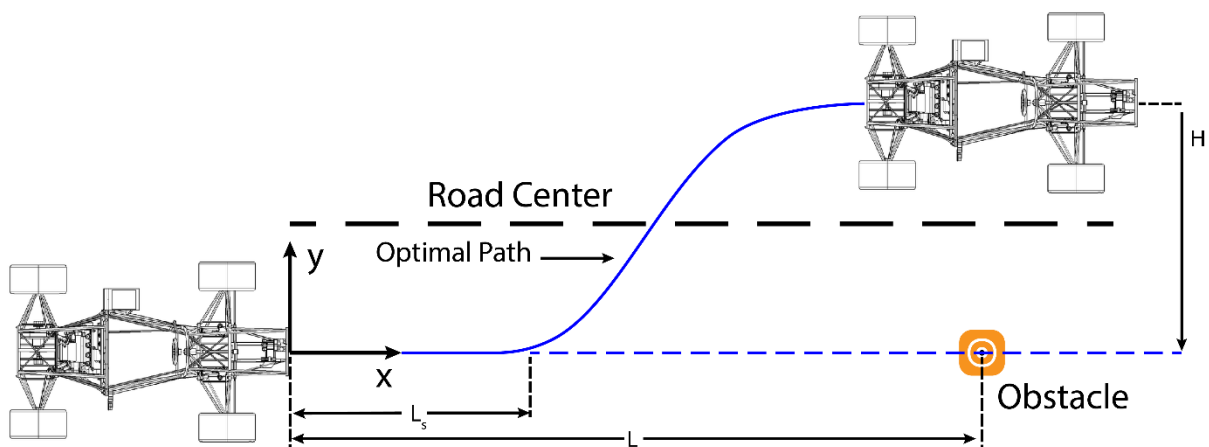


Figure 21. Lane change diagram



The input required to represent this maneuver in ADAMS/ View is using the STEP function. Figure 22 shows how a STEP function is defined in View. The Lane change is two STEP functions where the first ramps to the highest value while the second returns back to the initial value. The number of G's and how long they act are defined by the modeler.

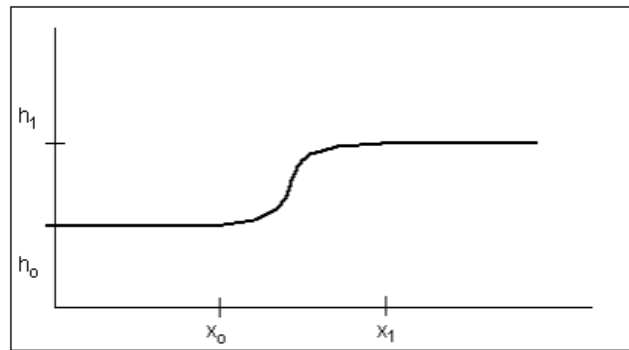


Figure 22. STEP function defined (5)

The equation format for the maneuver will look like the following:

$$\text{Lane Change} = \text{STEP}(X, x_0, h_0, x_1, h_1) - \text{STEP}(X, x_1, h_1, x_2, h_2)$$

Finally, when inputting the lane change equation, the number of G's inputted must equal the magnitude chosen. Therefore, the modeler must input the values in the respective X and Y windows that will equal the magnitude called for, as shown in Figure 23.

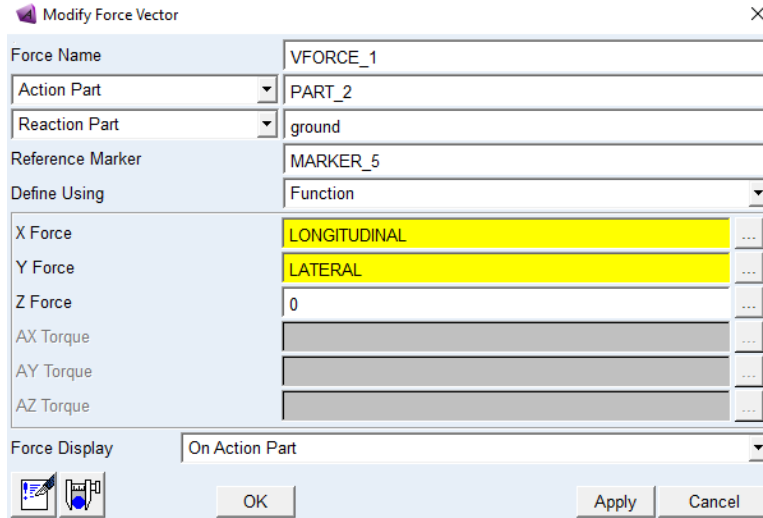


Figure 23. Force vector input window

X representing the longitudinal G's and Y the lateral G's. Keep in mind the direction the vectors assigned. The lateral does not matter as much as it defines either right or left but recall the longitudinal does as it either defines accelerating or braking.

### 2.3.1.2 Slalom

The slalom can be viewed as a sinusoidal wave in which the vehicle must weave between the obstacles laid before it in the fastest time possible.

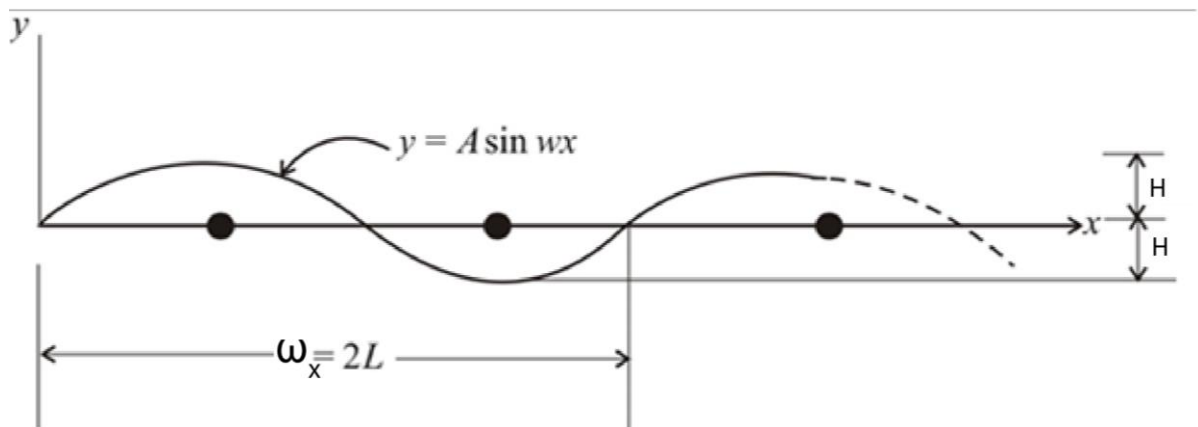
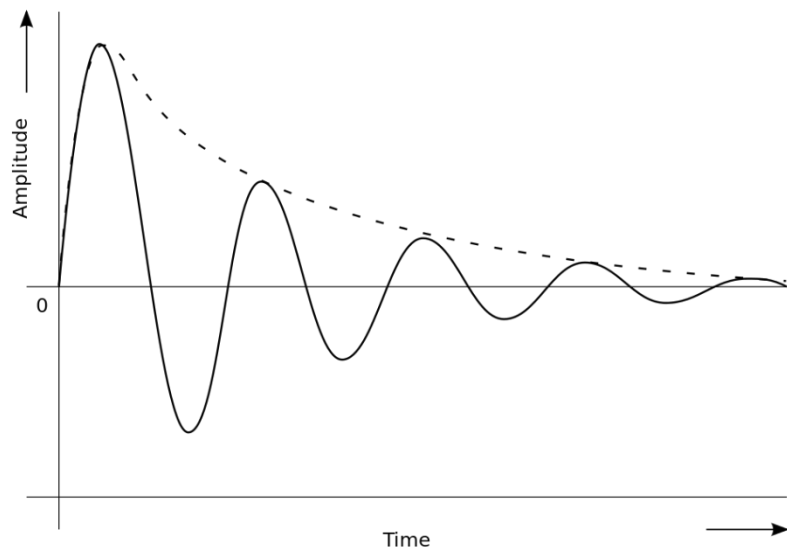


Figure 24. Slalom maneuver diagram

An input setup for the lateral is the SIN function. The SIN function inputs how a sine wave would behave, simply define the sinusoidal frequency and amplitude. An alternate input would be a damped sinusoidal wave, again laterally, shown in Figure 25. ADAMS/ View, at the time of writing this, does not have a damped sin wave function that can be used.



*Figure 25. Damped sinusoidal wave example*

Data from the UNM 2018 car will be used to produce a damped sine wave. A string of STEP functions could also be used to define a realistic slalom study. Again, it is the modelers responsibility to define the input so that it will correlate to actual data. The number of obstacles will define the number of STEP functions used. The use of damped sine waves to describe a slalom is described in sub-section 3.3.1.

### 2.3.2 Results

*The final step in the process. This step looks at pulling information from the simulations performed and analyzing what took place during those simulations.*

Once a simulation is successfully completed, a vast amount of data are available to the designer to explore. However, for the purposes of this thesis only a limited number of specific items will be considered. For the rigid model, its best to plot all the contact loads in the Z-direction. This is a similar plot to the equilibrium test but here the load transfer at the wheels is determined by the maneuver performed. The same plot can be made with the flexible model where torsional stiffness has a major impact. The respective plots should be compared to the rigid model plots for differentiation. The demonstration portion considers examples of actual plots that demonstrate the capability of the method.

### **3. APPLICATION OF PROCESS**

---

The demonstration follows the methodology in the same format as introduced, to demonstrate the application. The method developed will be applied to UNM's 2018 FSAE vehicle, shown in Figure 26. As a reminder, because the process is structured conceptually to encompass more systems, it is the modelers responsibility to make the necessary changes needed to represent their model. Some examples of these special circumstances will be presented here as well to provide a good example of where it can occur and how to address them.



*Figure 26. 2018 UNM LOBOMotorSports FSAE vehicle*

#### **3.1 Suspension**

The 2018 FSAE vehicle uses different front and rear shock configurations. The front has the shocks directly actuating from the wheel to the chassis whereas the rear has a push-rod + bell crank that actuates the shocks. Both the front and rear use the double wishbone design. Drop links attach from the bottom of the control arms to the

anti-roll bar. Figure 27 shows a corner of the front suspension while Figure 28 shows a corner from the rear suspension.

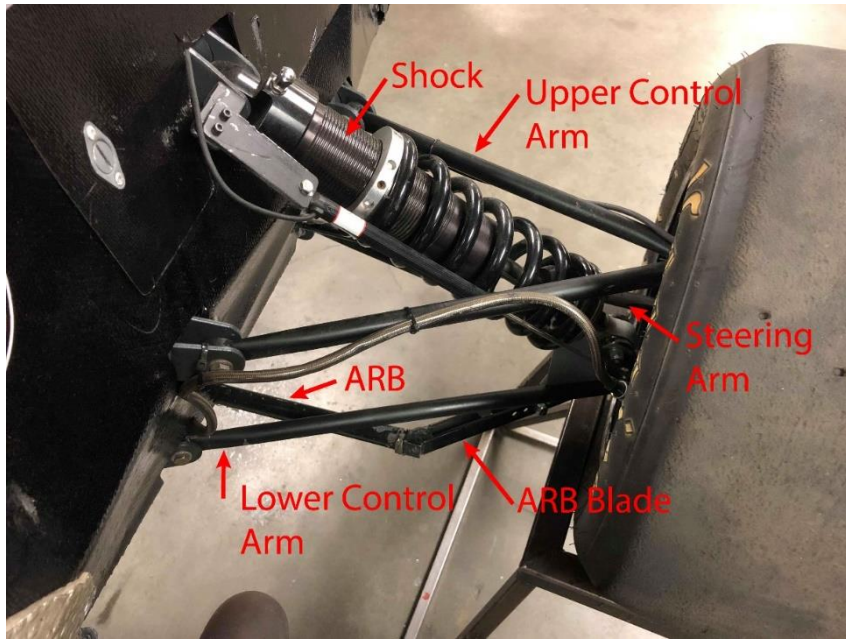


Figure 27. Front right corner of the 2018 FSAE vehicle

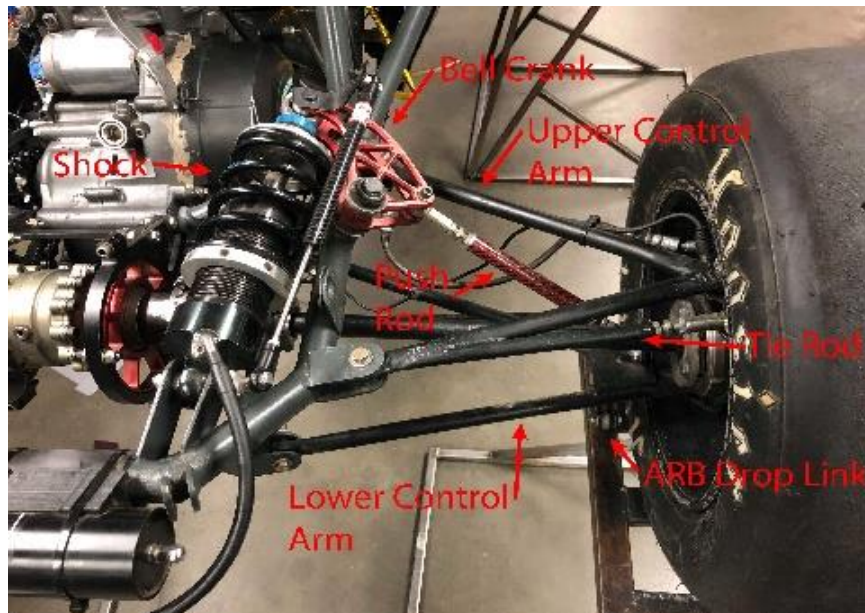


Figure 28. Rear right corner of the 2018 FSAE vehicle

### 3.1.1 Data Compiling

The data needed for the suspension was collected from the available documentation from UNM's LOBOmotorsports team records (see Appendix A). This will be referred to from here on. Note that the tables in Appendix A have the calculations for the front and rear weight distribution incorporated. A sample calculation is shown to verify points are within the criteria set in 2.1.2:

$$\text{Front Suspension Adjustment} = 62\text{in} * .48 = 29.76\text{in}$$

$$\text{Rear Suspension Adjustment} = 62\text{in} * 0.52 = 32.24\text{in}$$

#### *Wheelbase Verification*

$$\begin{aligned} &= |32.243\text{in}(\text{Front Hub Center}_x)| + |-29.76\text{in}(\text{Rear Hub Center}_x)| \\ &= 62.003\text{in} \approx 62\text{in} \checkmark \end{aligned}$$

$$\begin{aligned} \text{Front Track Width Verification} &= 24.809\text{in} (\text{Front Hub Center}_y) * 2 = 49.62\text{in} \\ &\approx 50\text{in} \checkmark \end{aligned}$$

$$\begin{aligned} \text{Rear Track Width Verification} &= 23.924\text{in} (\text{Rear Hub Center}_y) * 2 = 47.848\text{in} \\ &\approx 48\text{in} \checkmark \end{aligned}$$

### 3.1.2 Building Template

To begin building the template, following the methodology set, the parameters needed were readily referenced. Appendix A shows the suspension geometry adjusted to fit to the reference frame. The number of points needed were placed arbitrarily in space followed by adjusting accordingly. Figure 29 shows one side of the suspension created.

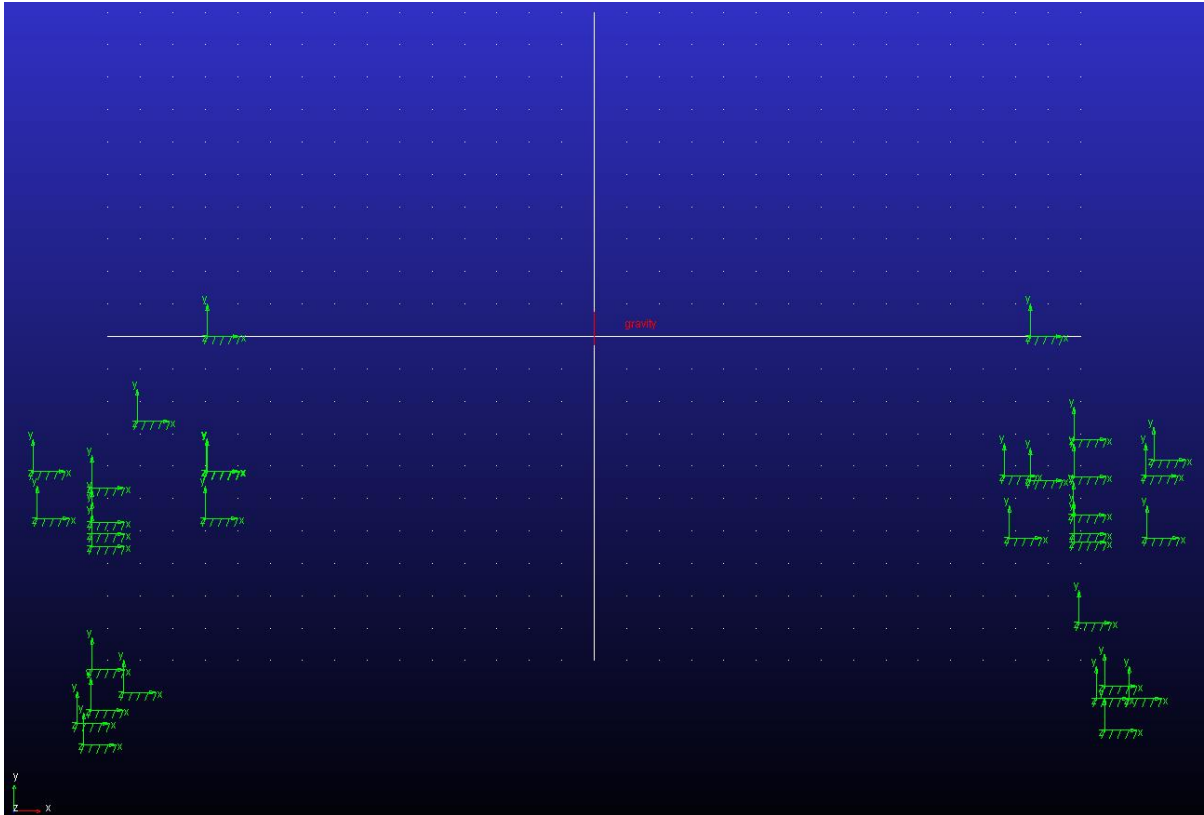


Figure 29. Right side of the template created

As the methodology mentions, the markers in the positive region of the X-coordinate, represent the front suspension points. Subsequently, the rear suspension points should be in the negative X-coordinate region. The naming convention used for these points are shown in Figure 30.



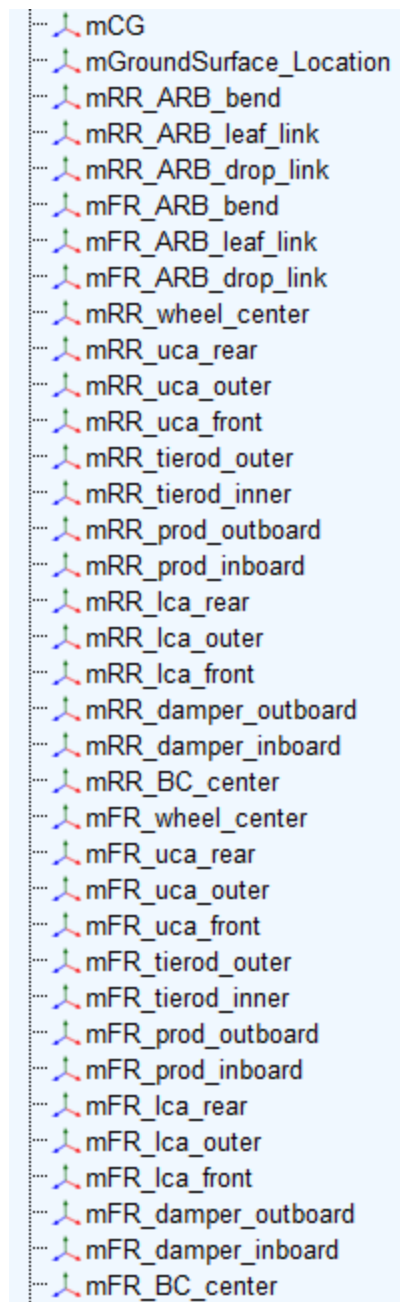


Figure 30. Naming Convention used for template

Since the right side of the suspension is used, Figure 30 depicts this as the third letter in the name indicates this. With one side of the suspension template complete, creating the other side is done by repeating the process of adding the correct number of markers with the respective names placed in space. Instead of inputting the left side

coordinates, parameterizing the template was done by using the mirror function. Figure 31 shows an example of how a left side marker is parameterized to a right-side marker in the location input.

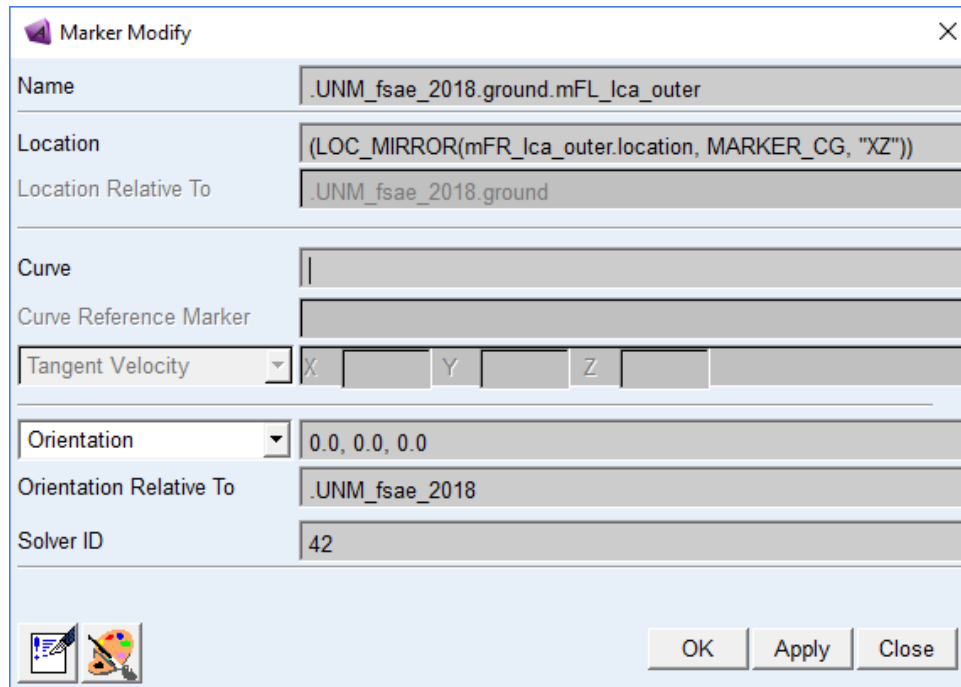


Figure 31. Mirror function depicted

Once all markers for the left side had been parameterized to the right-side suspension, a marker for the location of the CG is left to be placed. The marker should be located at the origin of the reference frame with only the z-coordinate as an input value. This value was 11in as shown in Appendix A while Figure 32 shows how this would look like. Figure 33 shows the completed template; ready for the next step.

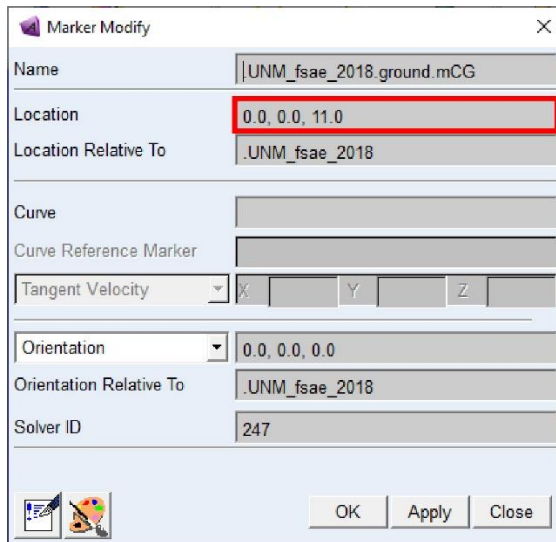


Figure 32. CG marker location

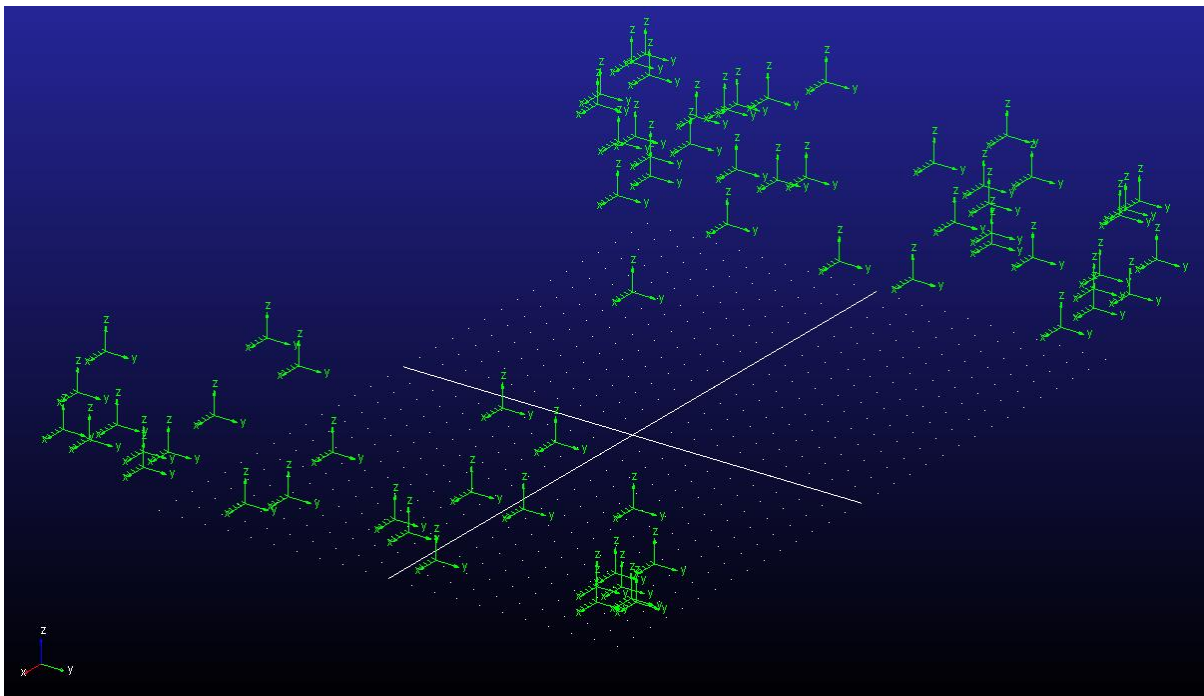
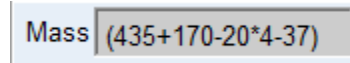


Figure 33. Full vehicle template complete

### 3.1.3 Creating the Parts

When creating the parts, as a reminder, keep the mass and inertia values in mind. A sphere was used to create the chassis of the vehicle at the CG template location. Four smaller sphere solids were added to the part where the shock connects to the chassis.

This will be explained in section 3.1.4. Lastly modifying the mass properties of the chassis part is shown in Figure 34.

A screenshot of a CAD software interface showing a mass property value. The text "Mass" is displayed in a light blue box, followed by the mathematical expression "(435+170-20\*4-37)" in a grey box.

*Figure 34. Vehicle mass value*

As mentioned in the methodology, even though the process for creating one corner of the vehicle at a time remains in concept, the process for creating the front and rear will be demonstrated for one corner each as again, the other side is mirrored. The order that was used in creating a corner is Bellcrank (if applicable) > Upright > Lower Control Arm > Upper Control Arm > P-rod (if applicable) > Tierod > Tire. Once a part had the necessary solids to define it, it was then renamed following the naming convention set in the methodology. Appendix C shows the naming convention used per part and its respective connections per part. After all corners had these components created, the anti-roll bar system was added in a similar process. Utilizing the template, it allowed solids to be created and attached with the right click feature. Appendix B assisted in determining what solids needed to be added to what group of part it needed to be. As an example, control arms are created with two-cylinder solids under one part. The only part that requires more detail work is the tires. The tires still use the cylinder solid but also use the features in the Bodies tab to further refine it. The “Fillet and edge” and “Hollow out a solid” was used to finalize the tires. Lastly, the steering rack was added. Once all parts were created and named, colors were assigned to identify components during simulation. Assigned mass values were inputted while inertial values were verified with their corresponding actual parts in CAD. Figure 35 depicts the front suspension components completed.

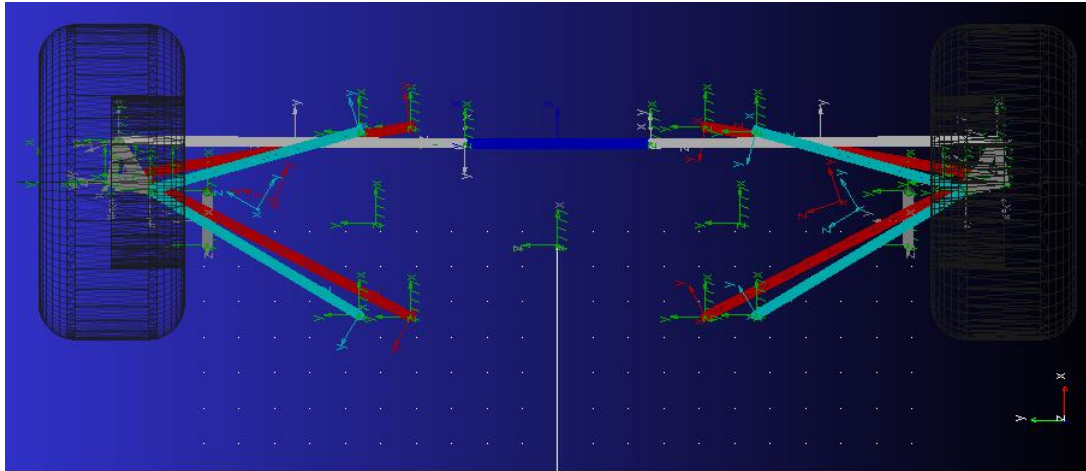


Figure 35. Front suspension parts added

For the rear, since the push rod + bell crank is utilized to actuate the shock, this adds more parts to the rear than the front. The tires in the rear follow the same process as the front, as well as most parts still utilizing the cylinder solid. Figure 36 depicts the rear suspension with the solids added and color coated.

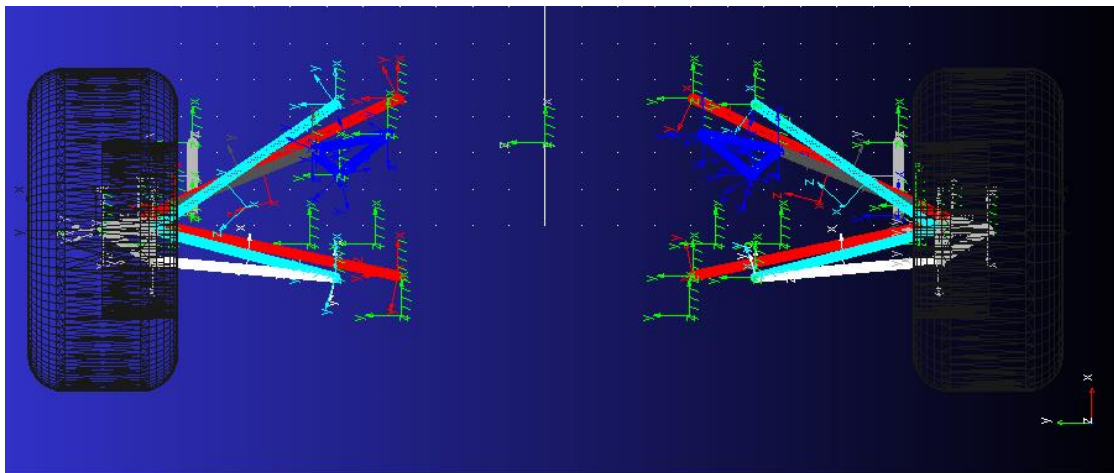
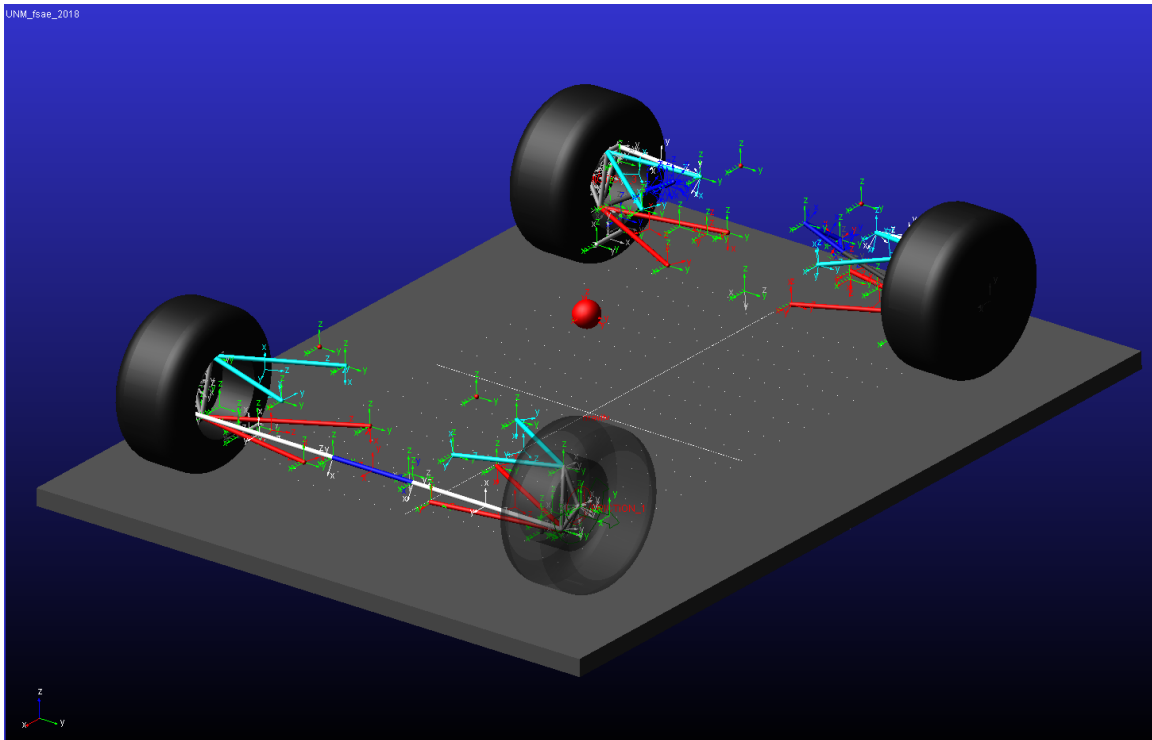


Figure 36. Rear suspension parts added

The last part needed is the ground or the track. The track is created using the box solid. The track was made big enough to cover the vehicle as well as have room to simulate. Finally, the solid was added to the ground since this part does not need to move or be connected to any components. Figure 37 shows the result of the

suspension model with all parts added. As an extra cautionary step, the revision of mass value per part. Certain parts calculated inertia was found to be satisfactory where others were not and were modified to represent such. Every part should be inspected before proceeding as one single part with an incorrect mass value can cause simulation errors. A good example is the tires. The tires from the cylinder alone, give a value of almost 400lbs, and the refinements do not take this into account, even if material was removed. The suspension now needs connections.



*Figure 37. Full suspension- parts added and complete*

### 3.1.4 Adding Connections

The joints necessary to define how the parts are interconnected to each other were identified. Appendix D lists the type joints used while also showing what parts are connected. Figure 38 depicts the model with joints added to the parts.

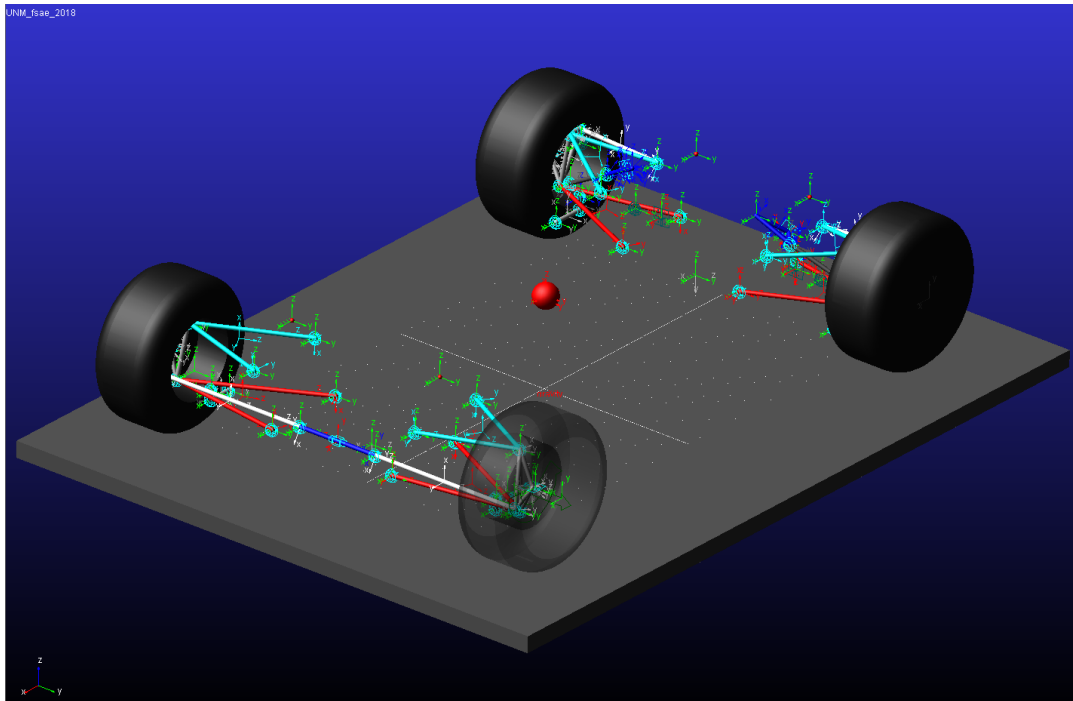


Figure 38. Full suspension- joints added

The next connections to add are the shocks. The shocks for the front were connected from the wheel to the chassis whereas the rear was connected from the bell crank to the chassis. Utilizing the four small spheres made for the chassis, allows the shocks to be connected to the chassis without any hassle. Once added, the spring rates were modified to the values in Appendix A. Figure 39 and 40 show the front and rear shocks respectively.

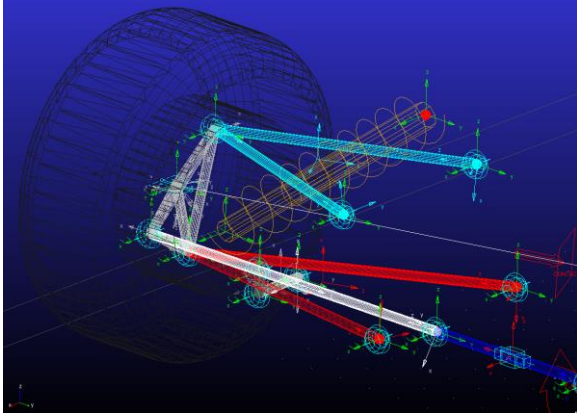


Figure 39. Front shock setup

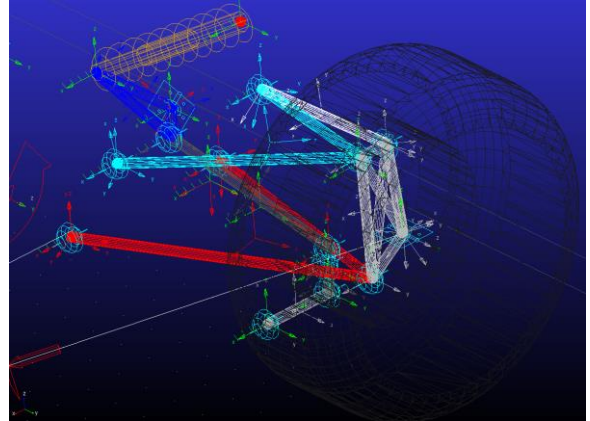


Figure 40. Rear shock setup

To add the ARB force component, use the torsion spring feature. The torsion spring should be connected between the left and right blade component with the location applied at the center of the respective front and rear suspension. Stiffness values were modified to the values on Appendix A. Figure 41 and 42 show the front and rear respectively.

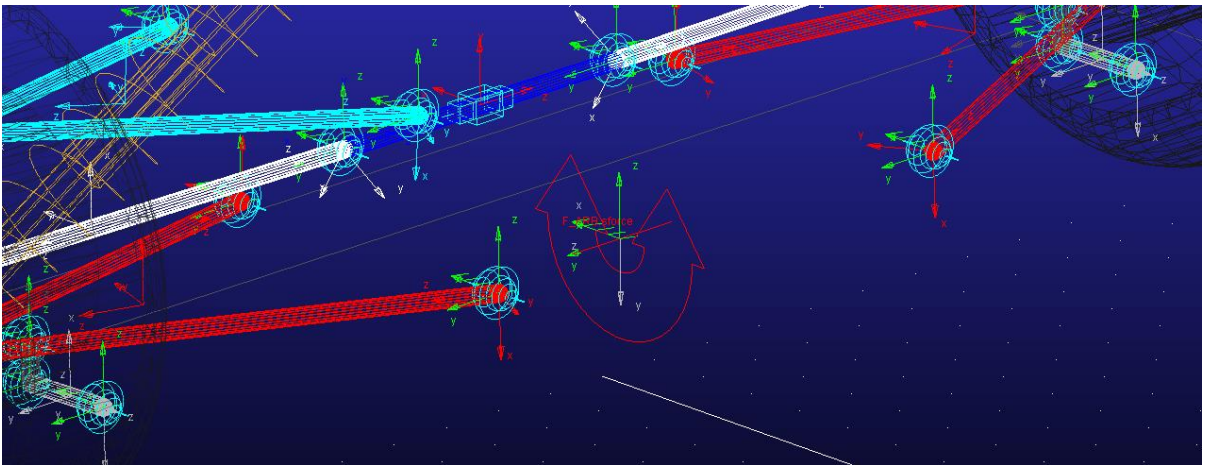


Figure 41. Front ARB torsion spring



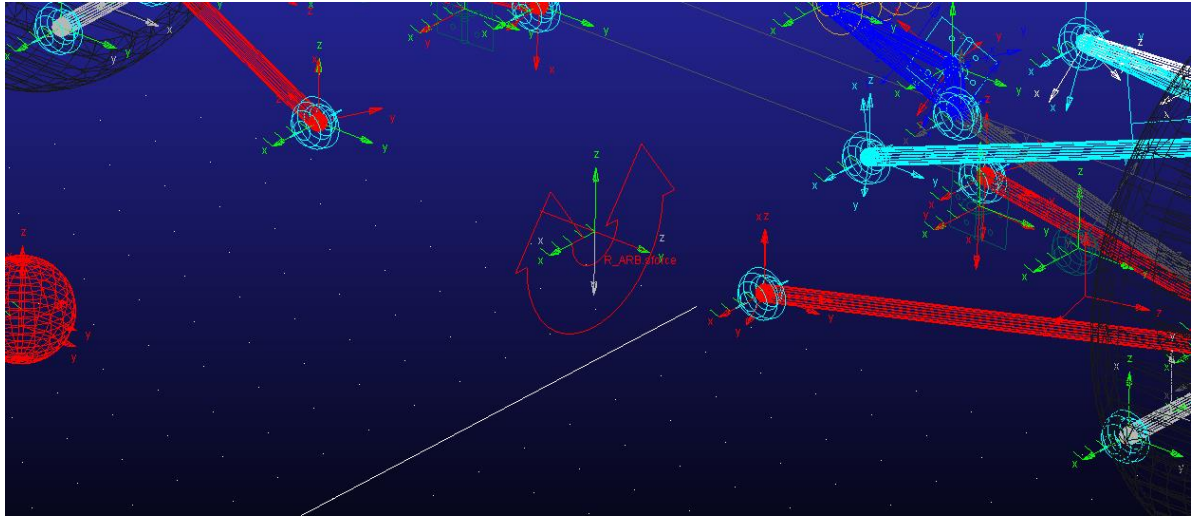


Figure 42. Rear ARB torsion spring

The contacts between the tires and ground must be established. Using the contacts feature, four contacts were created, one for each tire. The stiffness and damping values were modified to values that were more representative of actual tires, again a limitation to the lack of a tire model. These values are shown in Figure 43.

Stiffness	(179904(Newton/meter))
Force Exponent	2.2
Damping	(250(Newton-sec/meter))
Penetration Depth	3.937007874E-003

Figure 43. Tire stiffness and damping values used

The last connection needed is the applied force. The applied force is the force vector that will be used as the input for the model for defining the maneuvers to be performed. This force vector was created with the chassis as the action part while the ground as the reaction part, located at the CG marker. The suspension model is now completed and ready for verification.

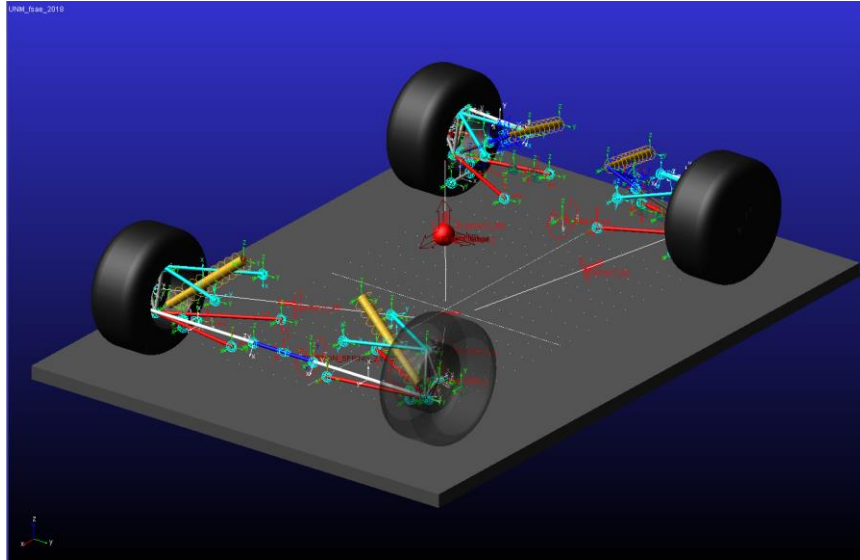


Figure 44. Full suspension- connections added, suspension model completed

### 3.1.5 Suspension verification

To verify the newly created suspension model, the equilibrium test was performed. Following the methodology, no input was given with the simulation window settings set to start from equilibrium.

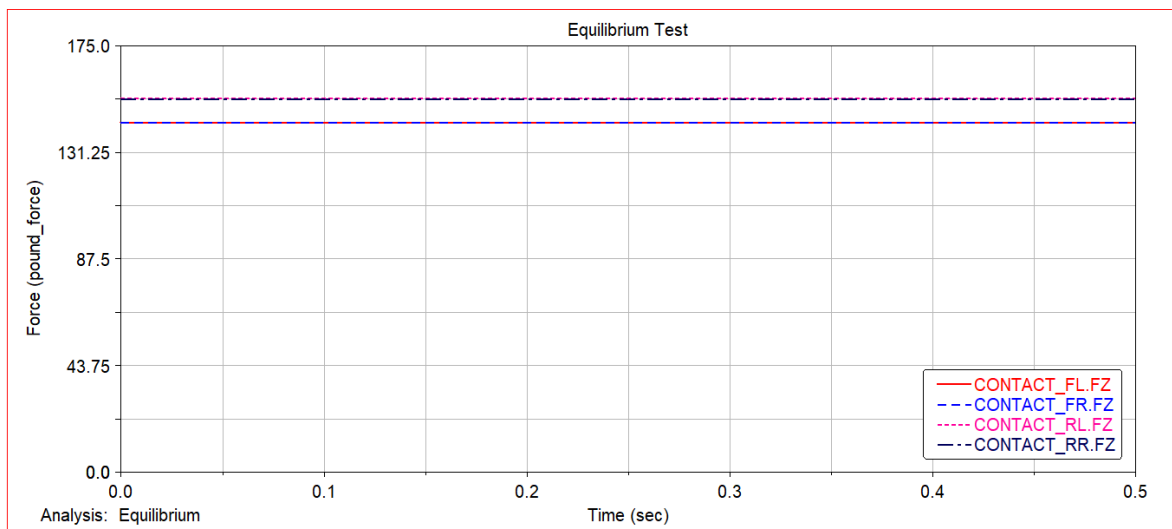


Figure 45. Equilibrium test – full scale

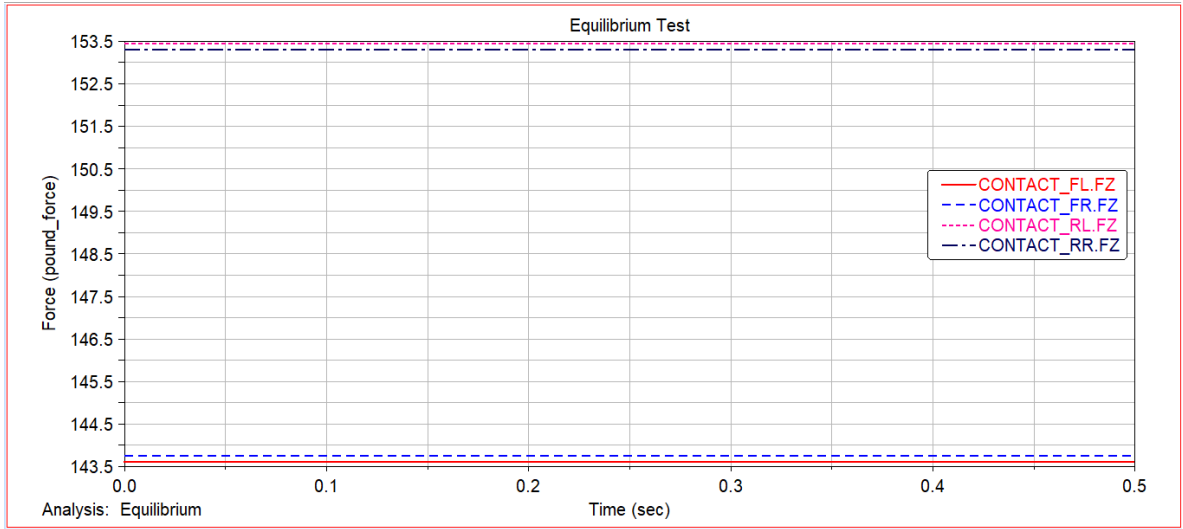


Figure 46. Equilibrium test – scale of interest

Note that Figure 45 and 46 depict the same results from the equilibrium test but with different limits in the vertical value. This was done to show an overall view followed by a focus view on the results themselves. Using Figure 46 and the plot tracking tool in the post processor, the values found at equilibrium are shown in Table 4.

Table 4. Equilibrium results per corner

Weight Distribution			
Front Left:	143.62 lbs	Front Right:	143.76 lbs
Rear Left:	153.45 lbs	Rear Right:	153.32 lbs

Verifying the values with the corresponding weight distribution, the following is found:

$$\text{Simulated Front}_{WD} = 143.76 \text{ lbs} + 143.62 \text{ lbs} = 287.38 \text{ lbs}$$

$$\text{Simulated Rear}_{WD} = 153.32 \text{ lbs} + 153.45 \text{ lbs} = 306.77 \text{ lbs}$$

$$\text{Theoretical Front}_{WD} = 605 \text{ lbs} * 0.48 = 290.4 \text{ lbs}$$

$$\text{Theoretical Rear}_{WD} = 605 \text{ lbs} * 0.52 = 314.6 \text{ lbs}$$

$$\text{Front \% Error} = \frac{|\text{Simulated Front}_{WD} - \text{Theoretical Front}_{WD}|}{\text{Theoretical Front}_{WD}} * 100 = 1.05\%$$

$$\text{Rear \% Error} = \frac{|\text{Simulated Rear}_{WD} - \text{Theoretical Rear}_{WD}|}{\text{Theoretical Rear}_{WD}} * 100 = 2.49\%$$

From the calculations above, the difference between the simulated and theoretical is within an acceptable margin of error. The acceptable limits set vary between modelers. However, for this demonstration, the error percentage is within an acceptable limit but can be further revised to reduce the error percentage. The suspension model is complete and verified, next is the addition of an accurate representation of the chassis.

### 3.2 Chassis

The chassis of the 2018 FSAE vehicle utilizes a space frame that adheres to the FSAE rules. Figure 48 and 49 show the fabricated chassis. This section shows the differentiation between the rigid and flexible model but again, follows the methodology of fully developing the rigid model first before attempting the flexible model.



Figure 47. Fabricated chassis rear view



Figure 48. Fabricated chassis front view

### 3.2.1 Chassis Preparations

The preparations for the chassis cover the setup for the model database in ADAMS/View. Figure 49 depicts the newly created database alongside with the recently created and verified suspension model database.

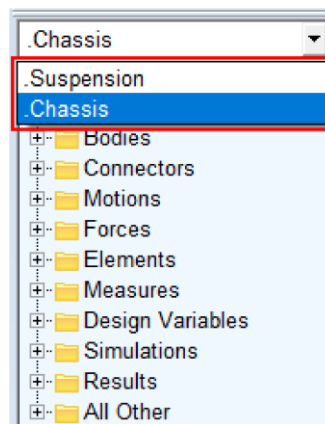


Figure 49. Chassis database

Note that when the new model was created, the same gravity settings were used since the chassis and suspension will be merged at a later point.

### 3.2.2 Data Compiling

The necessary CAD files were collected to import the space frame. As the methodology describes, the rigid model uses the STEP file, thus the CAD program had a version of the model created in that format. A final version of the chassis was used, any modifications to the chassis geometry or overall structure, should be made in the respective cad software as any modifications made in ADAMS/View should be towards the development of the full vehicle model in View. Additionally, the Parasolid file type was also collected so the flexible model could be created once ready to begin that process.

### 3.2.3 Importing

To import the CAD model for the chassis, Figure 50 depicts the window with the mentioned features selected to the new database model. Figure 51 depicts the imported cad model in the working space of View.

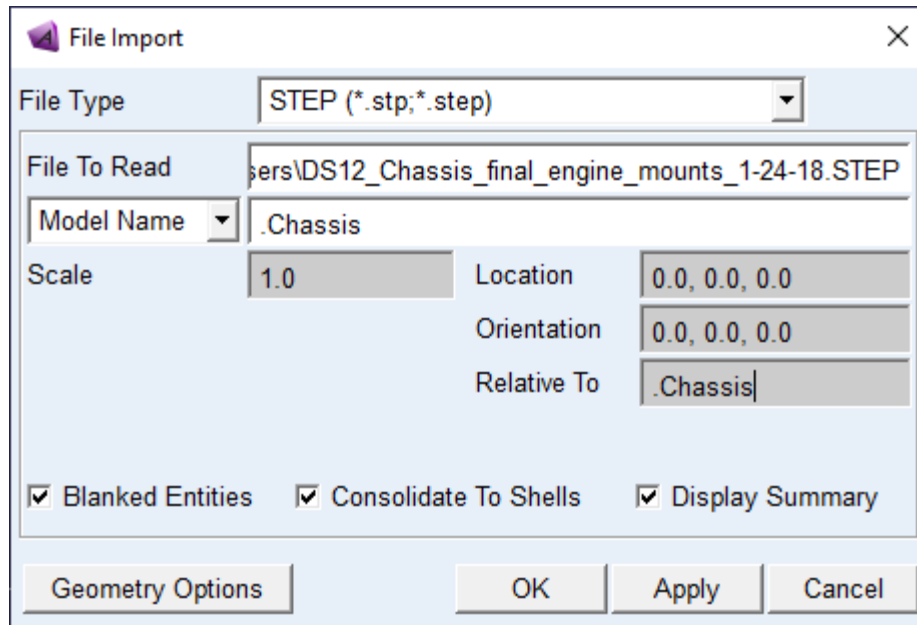


Figure 50. Import window setting configuration

The chassis CAD is imported with the color it was in the CAD software, in this case, it was a white tone. To differentiate what version the model will be, red will indicate the rigid model as this imported model was made. No further changes were required as the cad model used was the final version developed and used for the manufacturing of the prototype. No material properties were assigned as this won't be done until the next sub-section.

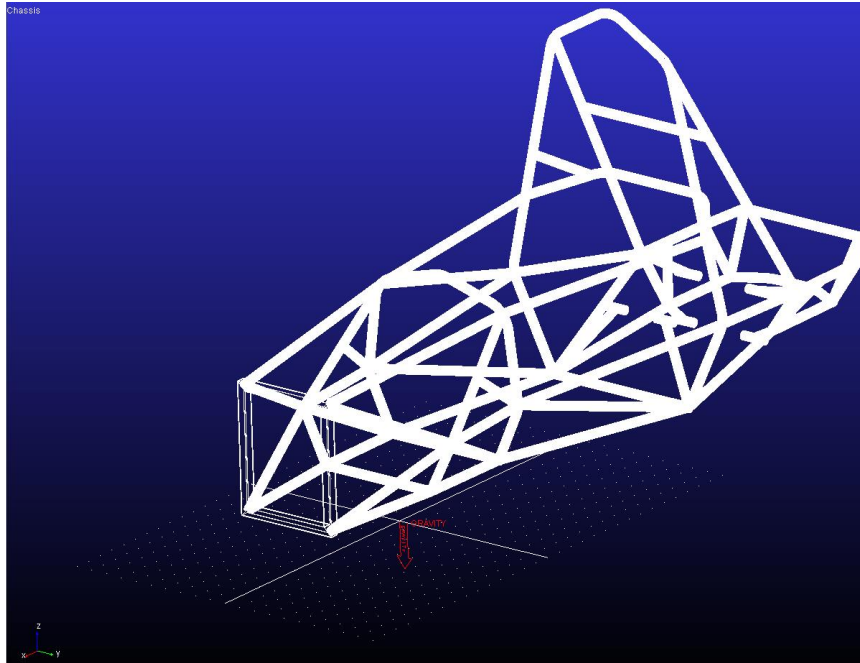


Figure 51. Imported cad of the chassis in ADAMS/ View

### 3.2.4 Model Merging

With the suspension and chassis model complete and ready to merge into one model; first is creating a new database. This new database is named UNM\_fsae\_2018, for the rigid full vehicle model. The suspension model is first merged to the UNM\_fsae\_2018 model as depicted in Figure 52.

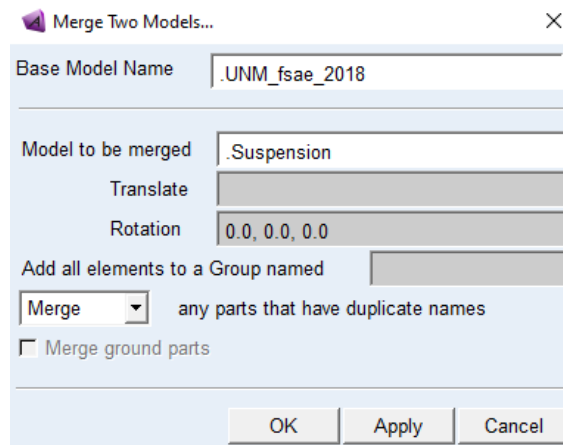


Figure 52. Merge window – suspension merged

The chassis follows a similar process with additional steps if the chassis model does not merge properly with the chassis part in the suspension model from the “merge any parts that have duplicate names” feature. If the feature does not perform as expected, all that is required is the chassis model to be merged with the suspension’s model chassis part using the Boolean feature (no-contact). The chassis model may be imported in the incorrect position, if so, identifying the respective marker that controls the geometry of the space frame to change its coordinates to reflect the correct position about the suspension model. Figure 53 shows the suspension and chassis model fully merged to be the new UNM\_fsaе\_2018 model. Material properties were then assigned to the space frame cad.

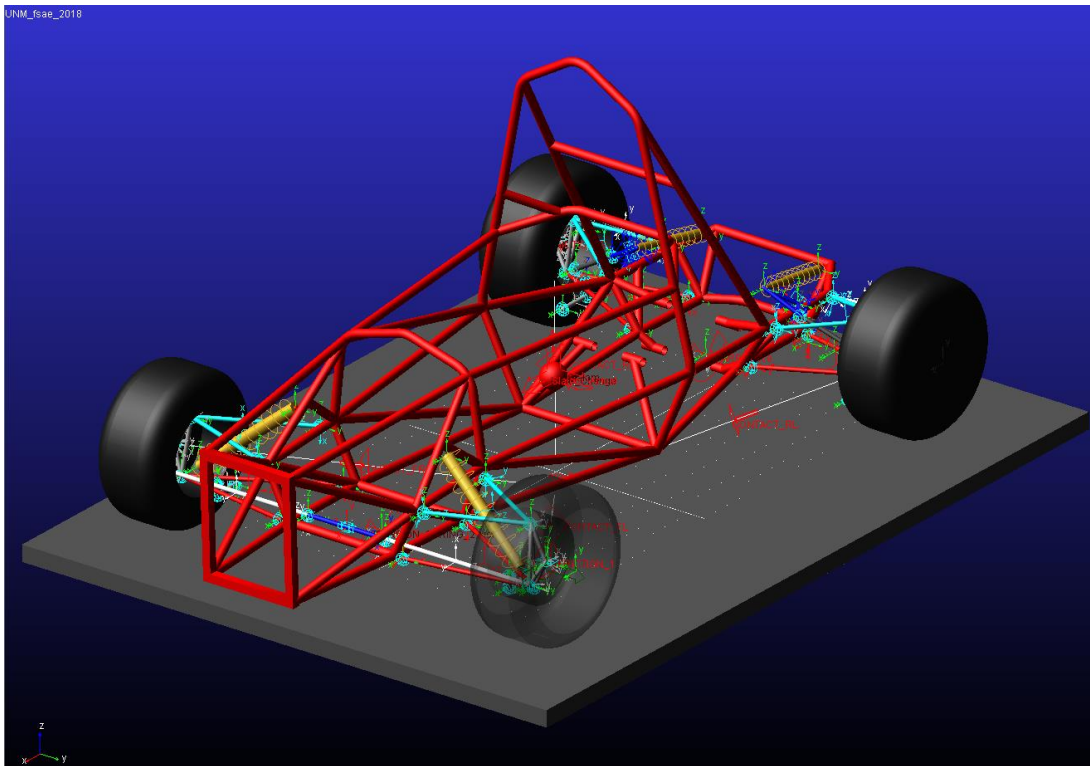


Figure 53. Fully merged model



### 3.2.5 ADAMS/ Flex

**IMPORTANT:** As stated, the flexible model was not created until the rigid model was complete and working properly in simulations. Any errors that occurred during the rigid model were addressed and resolved.

To create the full vehicle flexible chassis model, the process starts back in the importing of the chassis into ADAMS/ View. With the necessary file types collected, the Parasolid file type was used when importing the CAD model of the space frame. This is shown in Figure 54 with a new model database.

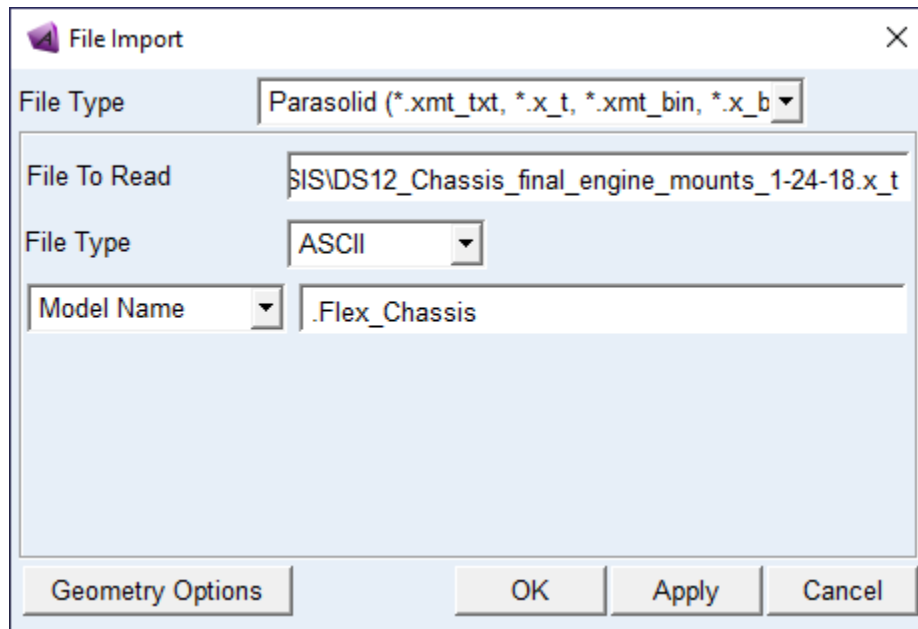


Figure 54. Importing of flexible compatible chassis cad

The importance of using the Parasolid is its compatibility with ViewFlex. Importing the Parasolid file type imports the space frame in component form instead of as a single part. This is shown in Figure 55 as almost 114 individual parts are imported with Figure 56 depicting a single solid highlighted within the imported components.

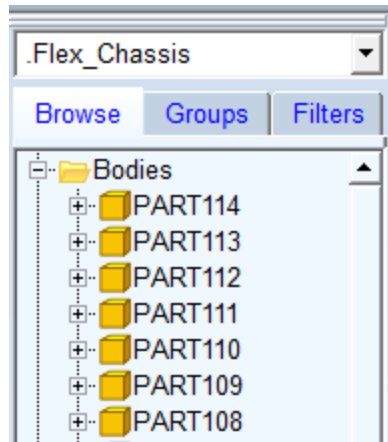


Figure 55. Flex chassis part quantity

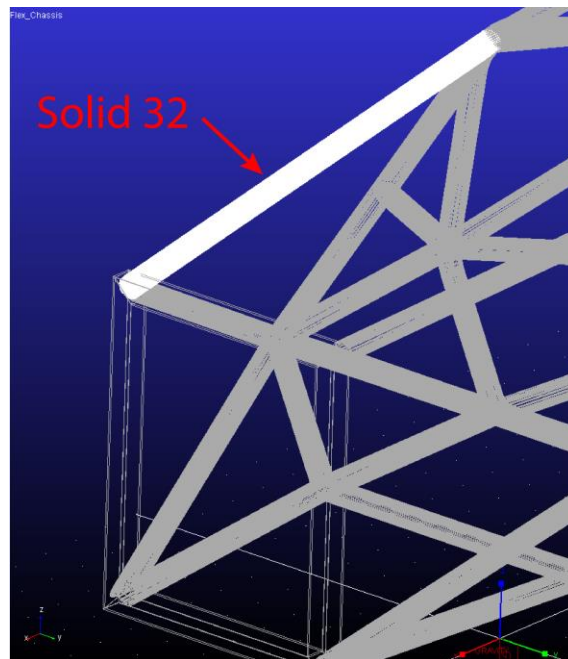


Figure 56. Single highlighted solid

With the space frame CAD imported, all the parts were united using the Boolean feature (in-contact) into one single part. Once all merged, the next step was to merge the Flex\_Chassis model with the suspension. A new model database was named Full\_Flex\_Model for the suspension and flexible chassis to be merged to. The same process used for merging the rigid model version was done as well. Once the flexible model is fully merged, additional steps are required before converting the space frame

to flexible. First off, the sphere that is used to represent the mass of the full vehicle needs to be separated from the chassis part. Therefore, this sphere is now its own new part fixed to the chassis, named “Vehicle Mass Rep” along with the provided mass values. Under the chassis part, only one solid should exist, the space frame as shown in Figure 57.

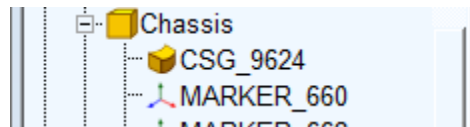


Figure 57. Single solid under Chassis part

The chassis is now one solid; therefore, it can be converted to a flexible part where Figure 58 depicts the settings used to convert the part using ViewFlex.

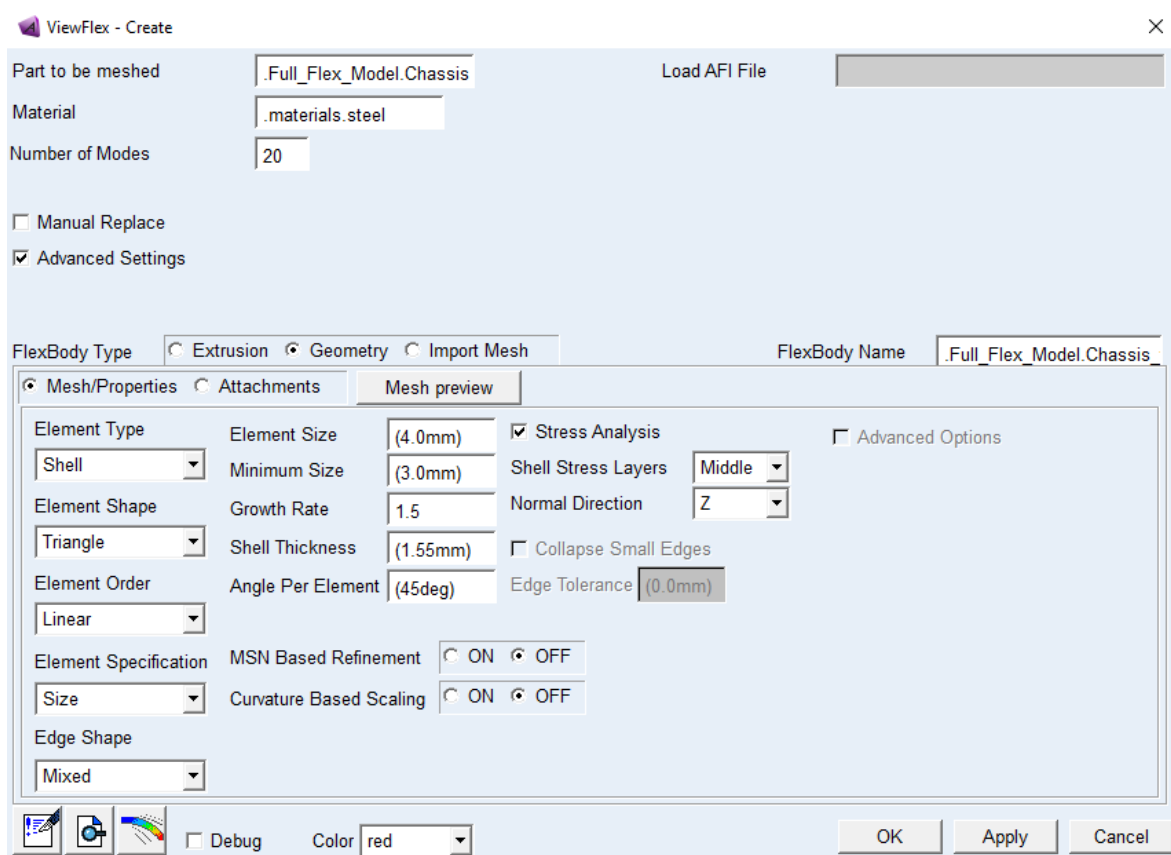


Figure 58. ViewFlex window configurations used

Once ViewFlex has created the mesh, and was deemed adequate, the last step was creating the attachment points by using the find attachments feature. Once the attachment table was populated, hitting “Ok” in the window started the process of replacing the space frame as a flexible body. Due to the limitation of available computer processing, the process took longer than expected. Once the conversion was complete, the chassis appearance was changed to dark grey and the flexible full vehicle model was complete, ready for simulation as depicted in Figure 59.

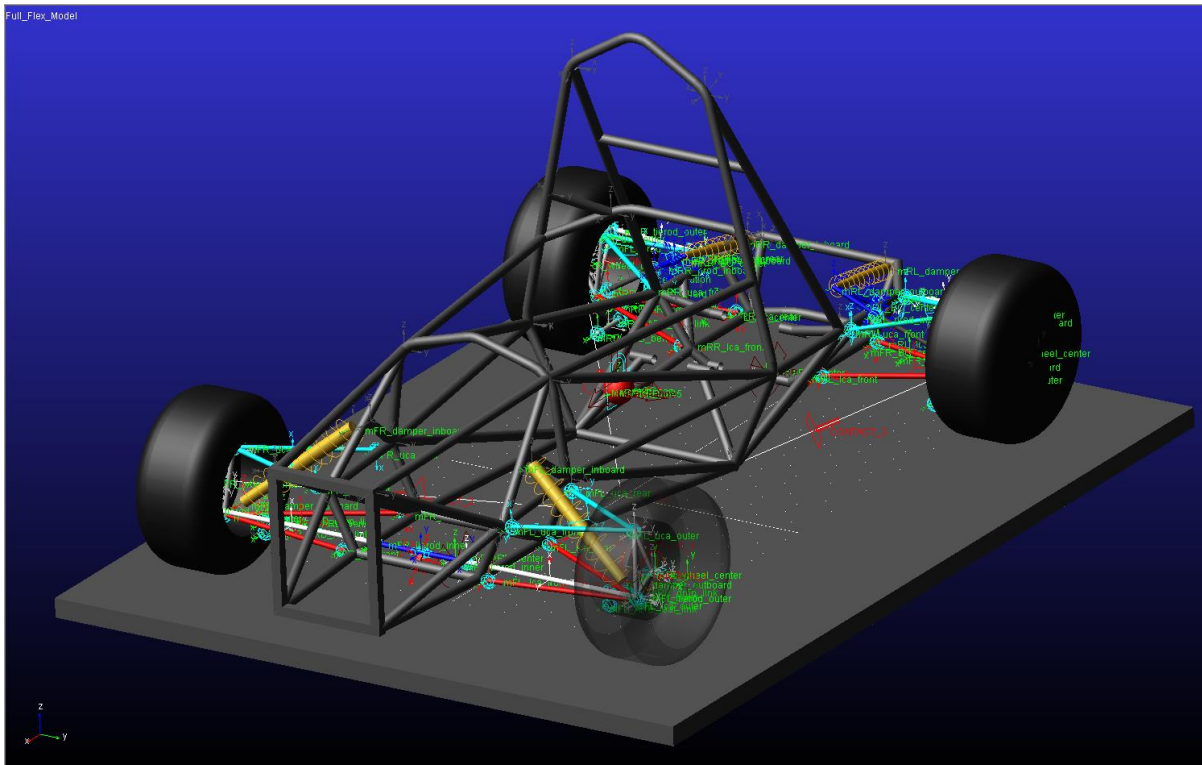


Figure 59. Full flexible model completed

### 3.3 Suspension + Chassis

Before proceeding with simulation with the completed full vehicle model, a study was performed on UNM 2018 to create semi-realistic simulations. Once the maneuvers were finalized and successfully simulated with the rigid model, the same exact

maneuvers were performed on the flexible model to compare results. A discussion portion was provided here as well as to provide a section of where the results can be reviewed and compared.

### 3.3.1 Simulation

Due to the method of input used for defining the simulations performed, it was necessary to perform a study on UNM 2018 car to create realistic simulations for the model. A GoPro was chosen to use the onboard video footage to capture a time study of the vehicle performing its maneuvers. The software used to extract the data was from Race Technology. A GoPro Hero 7 was mounted onto the top roll hoop of the prototype as rigidly as possible. The 2018 team proceeded to run the vehicle in a pre-defined course with driver swaps. This course is shown in Figure 60 with the course divided into sectors as well as color designation of runs made.

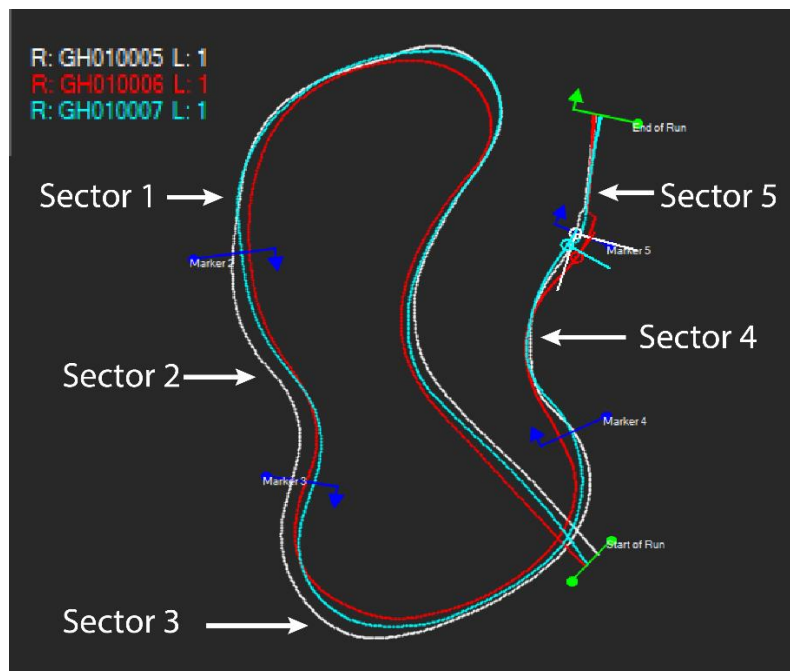


Figure 60. Course used and derived from Race Technology's software

Sector 2 was looked in detail due to its similarity to a lane change maneuver. Race Technology's software was used to determine the times as shown in Figure 61. The values shown were averaged to get a time of 2.687 seconds. Appendix E depicts plots with data from the three runs, all pertaining to sector 2. The video footage was able to create the time maneuvers are performed but lateral and longitudinal acceleration plots from Appendix E further refined the times.

Best	Sector 1	Sector 2	Sector 3	Sector 4	Sector 5
theoretical, 24.46	11.00	2.68	6.27	2.79	1.72
continuous, 24.46	11.00	2.68	6.27	2.79	1.72
simulated, 22.86	10.59	2.26	6.62	2.38	1.01
sim delta, 1.60	0.41	0.42	-0.35	0.41	0.71

Lap times	Sector 1	Sector 2	Sector 3	Sector 4	Sector 5
Run "GH010006" (33.71)					
Lap 1, 33.71	11.00 FC	2.70 C	6.34 C	4.10 C	9.57 C
Run "GH010005" (25.76)					
Lap 1, 25.76 F	11.77	2.68 F	6.80	2.79 F	1.72 F
Run "GH010007" (33.14)					
Lap 1, 33.14	11.93	2.68	6.27 F	3.28	8.98

Figure 61. Sector times per driver.

With the time study for the lane change complete, the maneuver was chosen to be performed in 3 seconds with longitudinal and lateral G's of 0.6 and 1.6 respectively. The data from Appendix E, though not verified, were found to be within realistic values and, as such, allowed to refine the equations for maneuvers further. Figure 62 depicts the equations used as input for the lane change simulation.

X Force	$(0.6 \cdot 605) \cdot (\text{STEP}(\text{time}, 0.0, 0.0, 1.5, 1.0) - \text{STEP}(\text{time}, 1.5, 0.0, 3.0, 1.0))$	...
Y Force	$(1.6 \cdot 605) \cdot (\text{STEP}(\text{time}, 0.0, 0.0, 1.5, 1.0) - \text{STEP}(\text{time}, 1.5, 0.0, 3.0, 1.0))$	...
Z Force	0	...

Figure 62. Equations of maneuver for lane change

The slalom event followed a similar procedure in which a smaller study, still using the GoPro method, was performed to understand how a slalom is defined from the UNM 2018 car. Originally, the sine function was to be used to define the slalom maneuver however, from the study conducted, it was found that the damp sine wave defined the maneuver more realistically. Figure 63 shows the equation used to define the lateral acceleration input for the model. The longitudinal acceleration was not of interest for this type of maneuver since the data from the footage show small affect in the longitudinal direction. It's important to note that the traditional slalom test was not used but instead, a slalom that would be seen in a typical FSAE course.

Define a runtime function 
 Full names  Short names  Adams ids

```

Step(time,0,0,0.4,(1.6*605))
+Step(time,0.4,0,1.2,(-(1.6*605)-(1.28*605))
+Step(time,1.2,0,2,(1.28*605+0.7*605))
+Step(time,2,0,2.36,(-0.7*(605)))

```

Figure 63. Lateral input for slalom maneuver

### 3.3.2 Results & Discussion

As previously discussed, this section reviews only the differences between the rigid and flexible model as well as the data available to the modeler/ user.

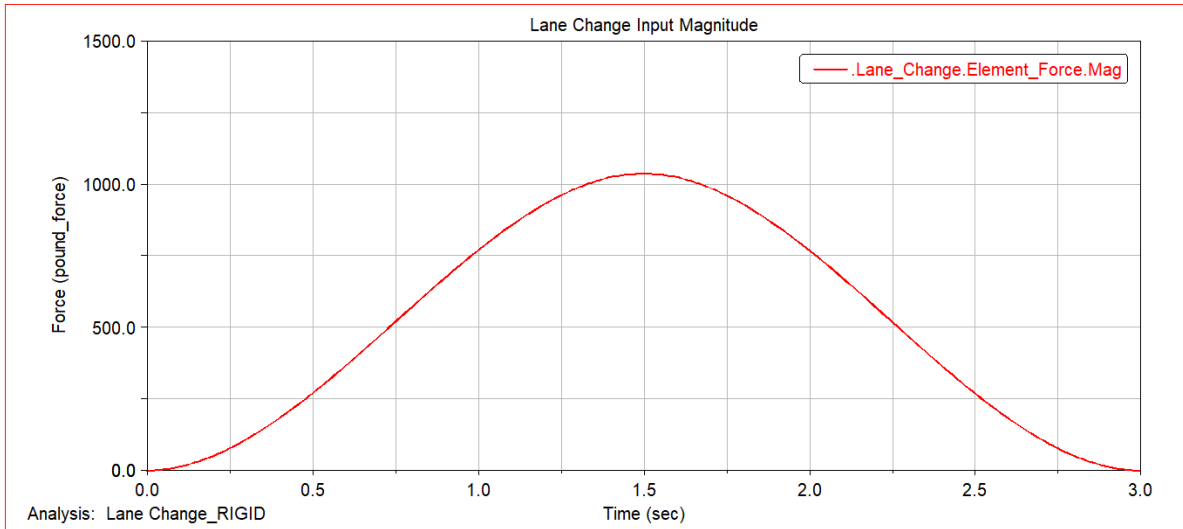


Figure 64. Lane Change input for rigid model

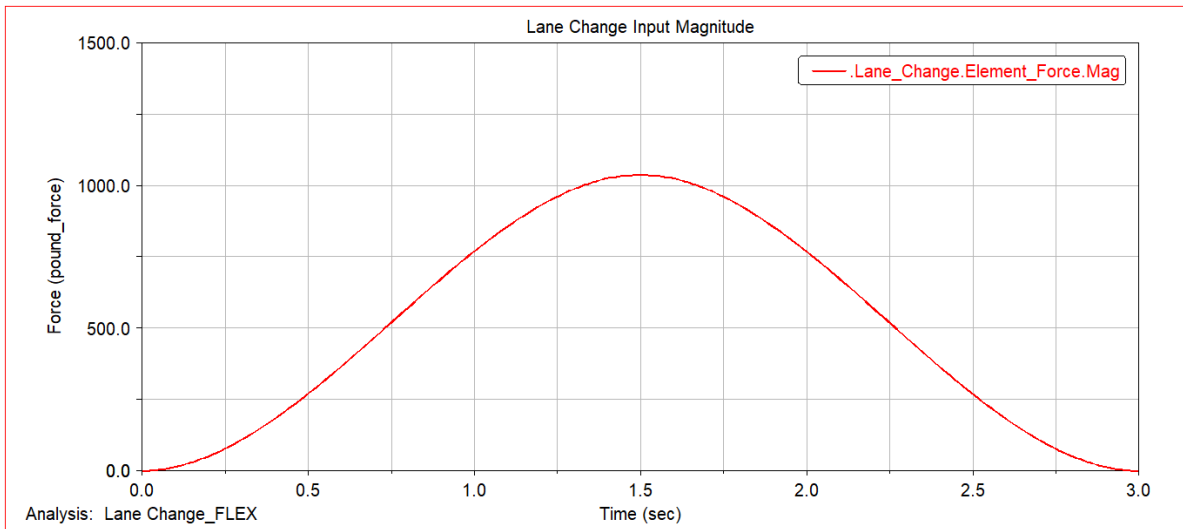


Figure 65. Lane Change input for flexible model

Figure 64 and 65 depict the magnitude of the input for the lane change for the rigid and flexible model respectively. This is to simply show that both models received the same input. Therefore, both the rigid and flexible see an overall load of 1.7 G's.



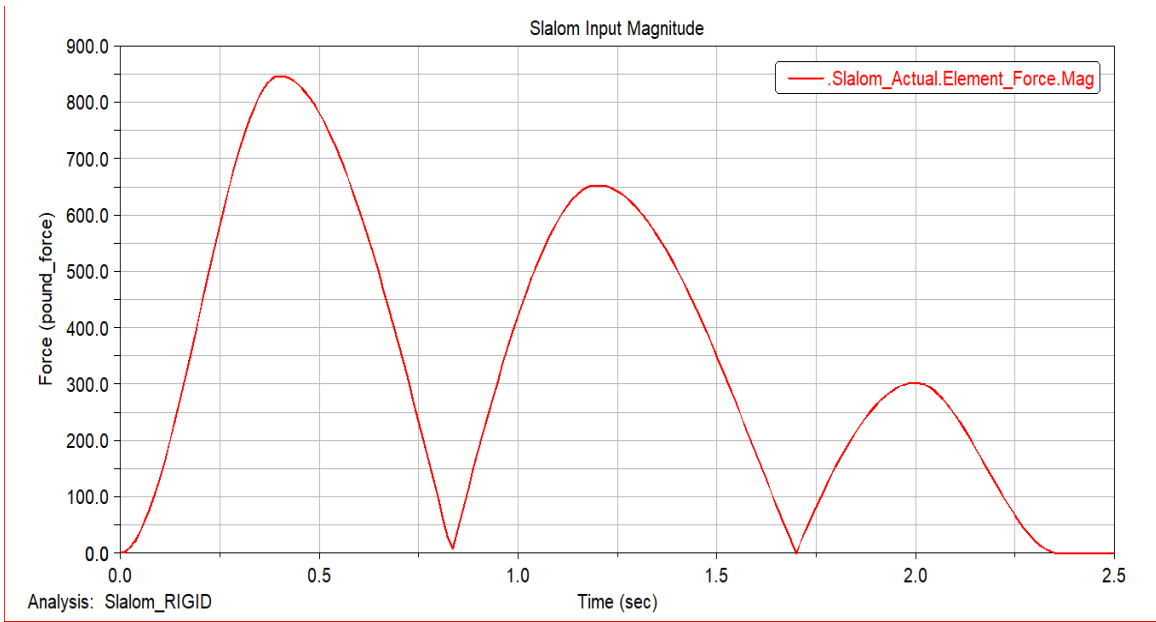


Figure 66. Slalom input for rigid model

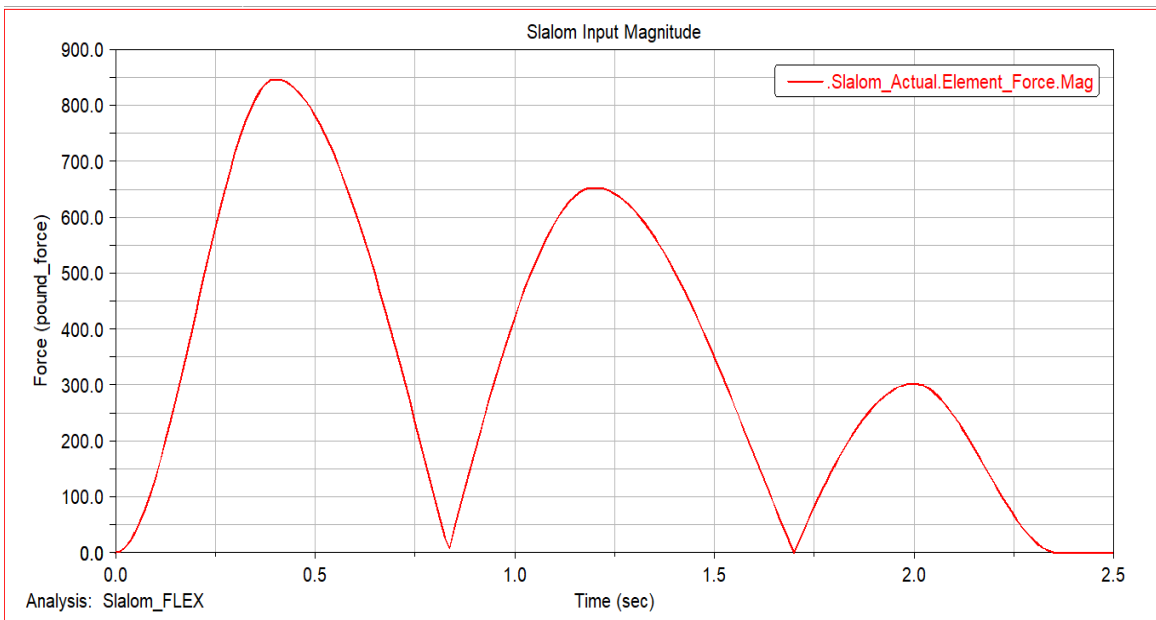


Figure 67. Slalom input for flexible model

A similar thing is viewable in Figure 66 and 67 where the input for the magnitude of the slalom is the same again for the rigid and flexible respectively.

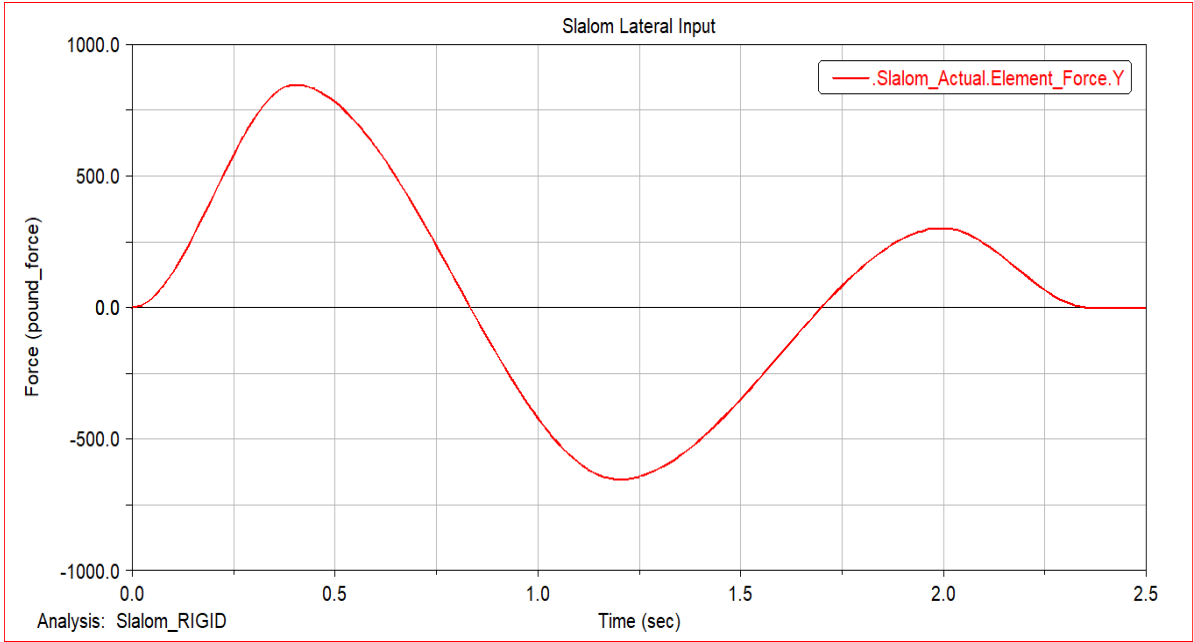


Figure 68. Slalom lateral input for rigid model

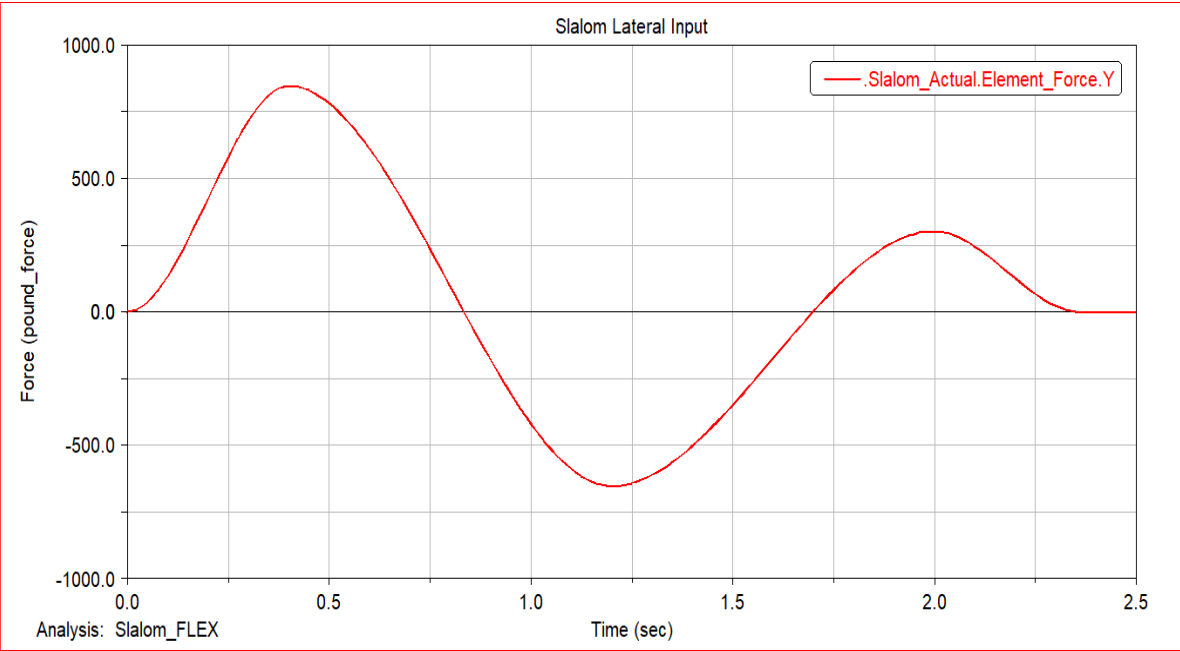


Figure 69. Slalom lateral input for flexible model

Figure 68 and 69 depict the lateral input for the rigid and flexible models respectively as is mainly shown to depict the defined behavior.

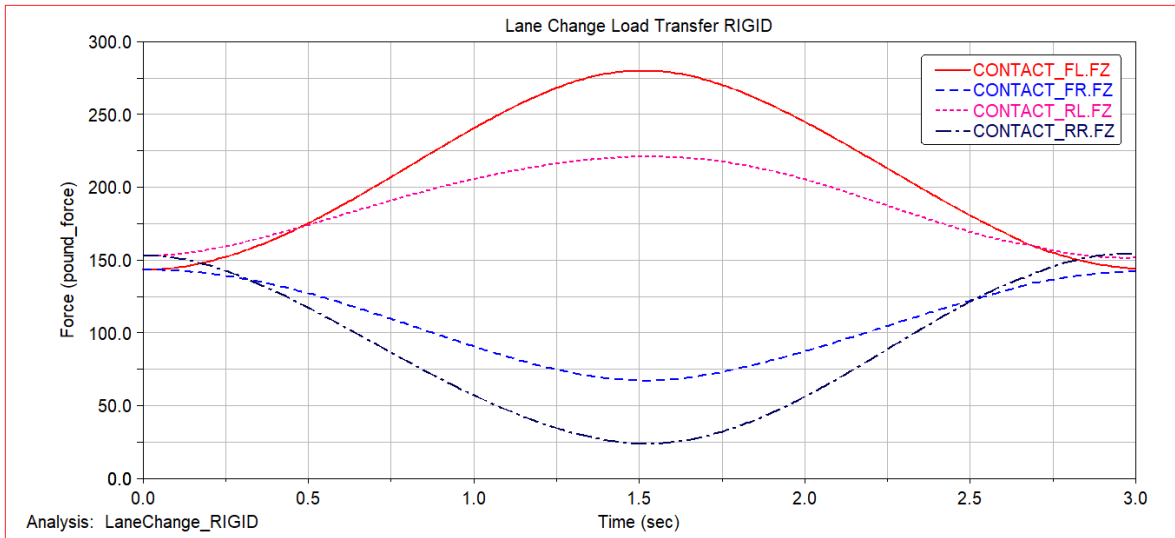


Figure 70. Lane change load transfer results for rigid model

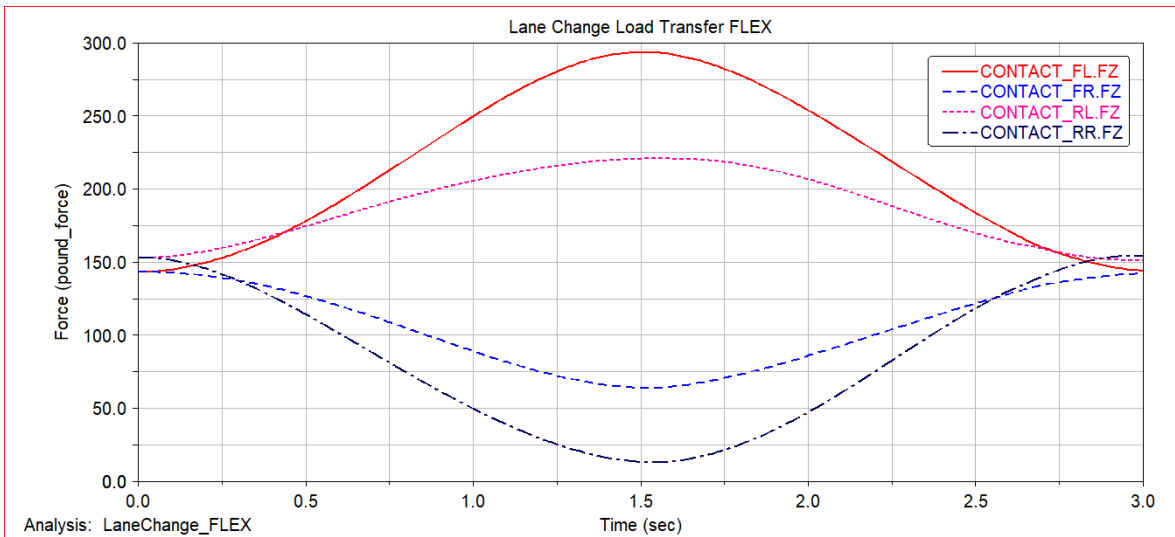


Figure 71. Lane change load transfer results for flexible model

The main results of the simulation are shown in Figure 70 and 71 where the rigid and flexible model's load transfer at the tires contact is shown for the lane change. Both the rigid and flexible share the same profile but with the flexible model incorporating the chassis as a flexible member, more load transfer can be seen. The difference between the two vary per corner. To better understand this, values at 1.5 seconds were found per corner per model as shown in Table 5.

Table 5. Lane change Load transfer values per corner

Lane Change Load					
Corner	Rigid	Units	Flex	Units	Difference (Flex - Rigid)
Front Left	280.36	lbs	294.33	lbs	13.97
Front Right	67.67	lbs	64.46	lbs	-3.21
Rear Left	221.38	lbs	221.21	lbs	-0.17
Rear Right	24.19	lbs	13.6	lbs	-10.59

According to Table 5, from the rigid to flex model, the front left gained approximately 14 pounds whereas the rear left saw almost no change. The front right similarly lost a small amount whereas the rear right lost a significant amount.

Looking at the slalom event, the results show a consistent change throughout the two models. Consistent in the difference between the left and right side of the vehicle's load transfer as to the lane change that had each corner vary significantly. Again, values were found to quantify the amount of disparity between the two models at 0.4 seconds or the peak of the load transfer.

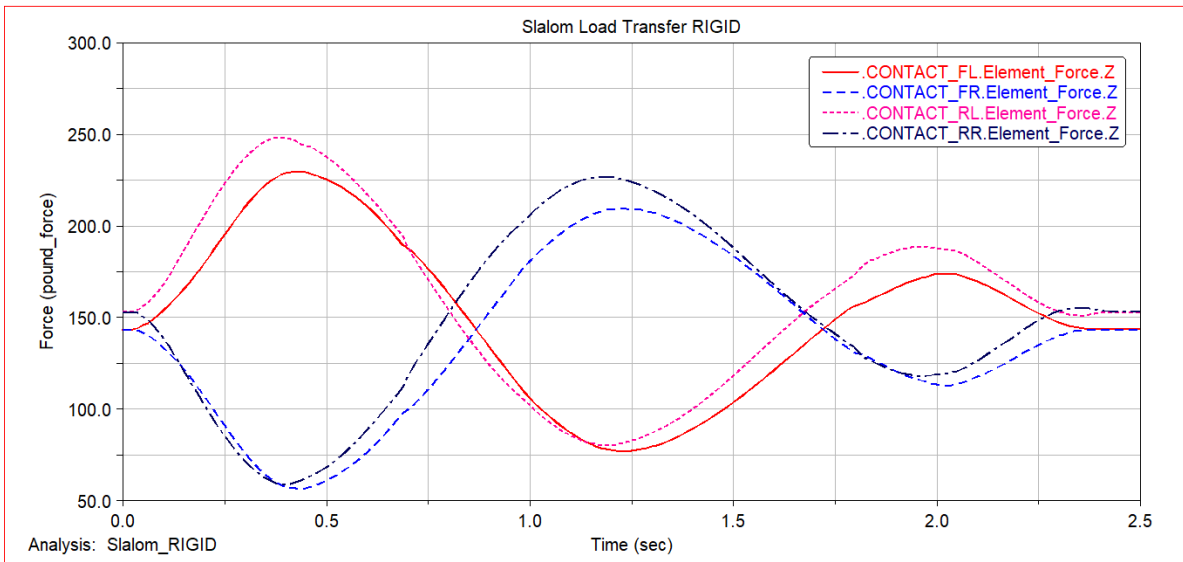


Figure 72. Slalom load transfer results for rigid model

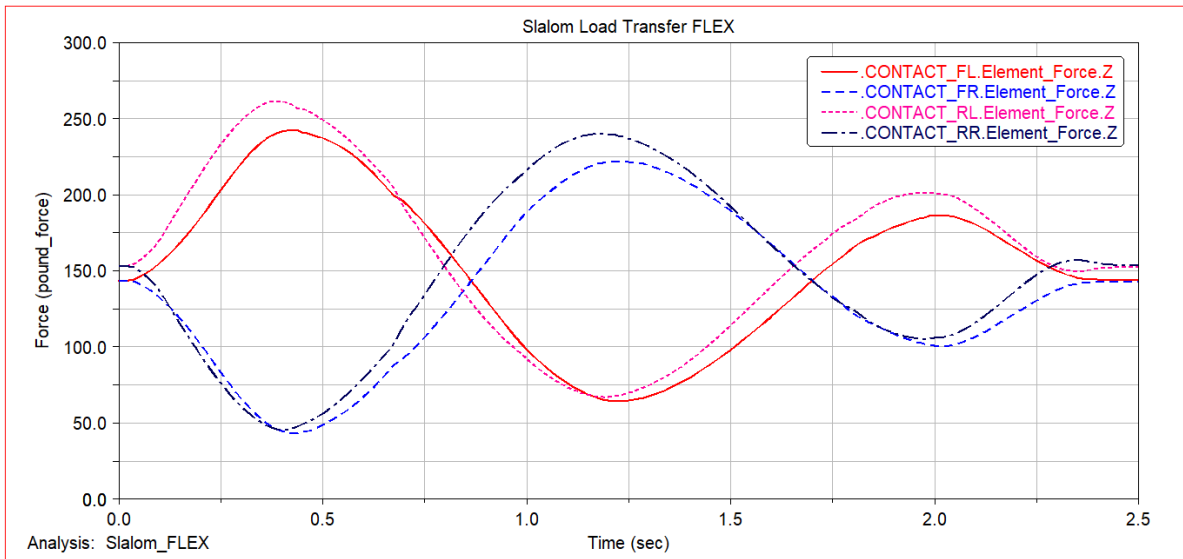


Figure 73. Slalom load transfer results for flexible model

Table 6. Slalom load transfer values per corner

Slalom Load					
Corner	Rigid	Units	Flex	Units	Difference (Flex - Rigid)
Front Left	228.76	lbs	241.64	lbs	12.88
Front Right	57.67	lbs	44.68	lbs	-12.99
Rear Left	248.44	lbs	261.55	lbs	13.11
Rear Right	58.89	lbs	45.89	lbs	-13

On average, the disparity between how much the flex model gained and lost was about 13 pounds as shown in Table 6. As mentioned, because the vehicle is approaching a turn, the lateral acceleration dampens out until returning to equilibrium.

#### **4. CONCLUSION**

---

The development of a new methodology that can provide analyses of the chassis with the incorporated suspension was established and described in detail so others can replicate it as no other method has been established. The results from the analyses performed proved useful, especially when comparing the rigid model to the flexible model. For the designer who is looking to begin studying or performing vehicle analysis, the introduction to this thesis will help those begin and provide a stepping stone into more complicated and complex analyses. Ultimately, with design becoming more and more prevalent in reducing cost and increasing efficiency, more than ever a multi-body dynamic analysis will help the designer get the edge they need to do it right the first time.

#### **5. FUTURE WORK**

---

During the development of the methodology, certain sub-processes were discovered and researched but were not implemented due to time constraints. These sub-

processes are shown in the Figure 2 as the alternate path. The addition of more data parameters at the cost of more computing resources will further the goal of having a model that will increase actual accuracy, but experimental studies must be done concurrently.

The following briefly covers what could be done and how these additions could benefit the methodology and the model(s).

- MNF File Process and Implementation

Modal Neutral File or MNF is the main file type that ADAMS uses in creating and storing the information needed to perform simulation and analysis of flexible components. The main issue concerned with creating MNF files is the process to create them is not readily available. Since MNF files are the main file type that are used to create flexible components but lack the information to create them, it's worth considering and creating a process that addresses this problem. As a starting point, ANSYS Mechanical APDL has been shown by other technical papers as software that is able to create/ communicate with ADAMS for MNF files. The benefit to this is the ability to create components with complex geometry and shapes that otherwise, ViewFlex is not able to mesh and create the necessary MNF file.

- Building Chassis in ADAMS (Only).

In the methodology presented, CAD that represents the chassis was imported instead of creating in ADAMS/ View in its entirety. ADAMS/ View body creation tab has enough tools to fully create, in this case, a space frame chassis. In other words, any members with different thicknesses and length or diameters, ADAMS/ View can

create. The benefit to creating the component in ADAMS/ View is when converting the part from rigid to flexible, the process is less computing resource extensive. What remains in question is if this process is worth the return versus just importing the CAD.

- Greater Tire Representation (Pacejka Formulas)

A highly beneficial addition to the model would be the Pacejka tire modeling equations. Instead of using a set value for the stiffness and damping of a tire, without a doubt, having the Pacejka formulas would greatly enhance the tire representation. Data from simulation could potentially prove much closer to actual tire behavior, which ultimately limit any vehicle, and thus be able to maximize their performance by optimizing the design. Other suspension parameters may be able to be used for as parameters such as toe and camber are not used in the methodology due again to simplification and replication.

- Instrumentation of Reference (actual) Model.

Experimental verification is needed to fully define the accuracy between the model and the prototype. Since the model has a large amount of data available for the user, it would be beneficial to establish a level of confidence at a more detailed level. The model results for example can be used for component sizing but lack an estimated accuracy. Ultimately, applying instrumentation to key areas of the actual prototype compared to the models created, will help identify any areas that require more refinement. At this time, it is assumed that the solution from the simulation that



ADAMS outputs are more conservative than what actual internal forces may be occurring.

## APPENDICES

### Appendix A. Suspension Parameters

<b>Suspension Design Parameters</b>	<b>Value</b>	<b>Units</b>
WheelBase	62	in
Front Track Width	50	in
Rear Track Width	48	in
Front Weight Distribution	48%	-
Rear Weight Distribution	52%	-
Front Spring Rate	425	lb/in
Rear Spring Rate	200	lb/in
Front ARB Stiffness	97.25	lb*ft/deg
Rear ARB Stiffness	108.1	lb*ft/deg
Vehicle Weight	435	lb
Driver Weight	170	lb
Suspension Weight	37	lb
Center of Gravity Height	11	in

ADAMS/View Suspension Points						
Front	Left			Right		
	x	y	z	x	y	z
Lower A-arm forward	35.322	8.2	5.755	35.322	-8.2	5.755
Lower ball joint	32.64	23.515	5.605	32.64	-23.52	5.605
Lower A-arm rearward	24.773	8.2	5.755	24.773	-8.2	5.755
Upper A-arm forward	35.069	11.05	11.688	35.069	-11.05	11.688
Upper ball joint	31.814	22.7	12.295	31.814	-22.7	12.295
Upper A-arm rearward	24.861	11.05	11.688	24.861	-11.05	11.688
Steering tie-rod on hub	34.555	24.43	6.85	34.555	-24.43	6.85
Steering tie-rod inboard	34.39	5.157	6.75	34.39	-5.157	6.75
Hub center	32.243	24.809	8.748	32.243	-24.81	8.748
Caliper brake pad center	28.665	24.809	8.748	28.665	-24.81	8.748
Spring upper on CHASSIS	29.988	10.126	16	29.988	-10.13	16
Spring lower on SUSPENSION	32.25	21.44	7.15	32.25	-21.44	7.15
Anti-roll bar axis	28.74	19.5	4.255	28.74	-19.5	4.255
Anti-roll bar mid-point	28.74	0	4.255	28.74	0	4.255
Anti-roll bar blade	31.74	19.5	4.2	31.74	-19.5	4.2
Anti-roll bar attach	31.74	19.5	5.4	31.74	-19.5	5.4
Rear	Left			Right		
	x	y	z	x	y	z
Lower A-arm forward	-22.67	7.976	5.625	-22.67	-7.976	5.625
Lower ball joint	-29.26	22	5.77	-29.26	-22	5.77
Lower A-arm rearward	-32.25	7.976	5.625	-32.25	-7.976	5.625
Upper A-arm forward	-22.98	11.358	10.809	-22.98	-11.36	10.809
Upper ball joint	-29.76	21.25	11.92	-29.76	-21.25	11.92
Upper A-arm rearward	-32.34	11.358	10.809	-32.34	-11.36	10.809
Steering tie-rod on hub	-31.44	21.25	11.92	-31.44	-21.25	11.92
Steering tie-rod inboard	-32.34	11.358	10.809	-32.34	-11.36	10.809
Hub center	-29.76	23.924	8.75	-29.76	-23.92	8.75
Pivot Axis Midpoint	-26.81	11.13	12.121	-26.81	-11.13	12.121
Spring upper on CHASSIS	-34.38	7.771	12.259	-34.38	-7.771	12.259
Spring lower on SUSPENSION	-24.57	8.49	14.025	-24.57	-8.49	14.025
Caliper brake pad center	-26.72	23.924	8.75	-26.72	-23.92	8.75
Push/Pull rod pivot	-25.44	12.573	11.65	-25.44	-12.57	11.65
Push/Pull rod attach	-28.44	19.994	7.021	-28.44	-19.99	7.021
Anti-roll bar axis	-25.03	19.072	4	-25.03	-19.07	4
Anti-roll bar mid-point	-25.03	0	4	-25.03	0	4
Anti-roll bar blade	-29.01	19.072	4	-29.01	-19.07	4
Anti-roll bar attach	-29.01	19.072	5.5	-29.01	-19.07	5.5

## Appendix B. PART CREATION GUIDE

Part Creation Guide				
Part	Solid Type	Quantity	Features	Color
CG	Spherical	4	N/A	Red
Bellcrank	Cylinder	3	N/A	Blue
Upright	Cylinder	6	N/A	Silver
LCA	Cylinder	2	N/A	Red
UCA	Cylinder	2	N/A	Cyan
P-Rod	Cylinder	1	N/A	Grey
Tierod	Cylinder	1	N/A	White
Tire	Cylinder	1	Fillet Hollow	Black
DropLink	Cylinder	1	N/A	Grey
Blade	Cylinder	1	N/A	White
Steering Rack	Cylinder	1	N/A	Blue

## Appendix C. ADAMS 2018 FSAE MODEL STRUCTURE (By Parts)

Topology of model: UNM\_fsae\_2018

Ground Part: ground

Part ground

Is connected to:

FL_Tire	via CONTACT_FL	(Contact)
RL_Tire	via CONTACT_RL	(Contact)
FR_Tire	via CONTACT_FR	(Contact)
RR_Tire	via CONTACT_RR	(Contact)
Chassis	via Slalom	(Force_Vector)
Chassis	via Lane_Change	(Force_Vector)
Chassis	via Slalom_Actual	(Force_Vector)

Part Chassis

Is connected to:

FR_BC	via J_FR_BC_center	(Revolute Joint)
FR_LCA	via FR_Shock.sforce	(Single_Component_Force)
FR_LCA	via J_FR_LCA_front	(Spherical Joint)
FR_LCA	via J_FR_LCA_rear	(Spherical Joint)
FR_UCA	via J_FR_UCA_front	(Spherical Joint)
FR_UCA	via J_FR_UCA_rear	(Spherical Joint)
RR_BC	via J_RR_BC_center	(Revolute Joint)
RR_BC	via RR_Shock.sforce	(Single_Component_Force)
RR_LCA	via J_RR_LCA_front	(Spherical Joint)
RR_LCA	via J_RR_LCA_rear	(Spherical Joint)
RR_Tierod	via J_RR_Tierod_inner	(Spherical Joint)
RR_UCA	via J_RR_UCA_front	(Spherical Joint)
RR_UCA	via J_RR_UCA_rear	(Spherical Joint)
FR_ARB_Blade	via J_FR_ARB_Bend	(Spherical Joint)
RR_ARB_Blade	via J_RR_ARB_Bend	(Spherical Joint)
FL_LCA	via FL_Shock.sforce	(Single_Component_Force)
RL_BC	via RL_Shock.sforce	(Single_Component_Force)
FL_BC	via J_FL_BC_center	(Revolute Joint)
FL_UCA	via J_FL_UCA_front	(Spherical Joint)
FL_UCA	via J_FL_UCA_rear	(Spherical Joint)
FL_LCA	via J_FL_LCA_front	(Spherical Joint)
FL_LCA	via J_FL_LCA_rear	(Spherical Joint)
Steering_Rack	via J_Steering	(Translational Joint)
RL_BC	via J_RL_BC_center	(Revolute Joint)
RL_UCA	via J_RL_UCA_front	(Spherical Joint)
RL_UCA	via J_RL_UCA_rear	(Spherical Joint)
RL_LCA	via J_RL_LCA_front	(Spherical Joint)
RL_LCA	via J_RL_LCA_rear	(Spherical Joint)
RL_Tierod	via J_RL_Tierod_inner	(Spherical Joint)
FL_ARB_Blade	via J_FL_ARB_Bend	(Spherical Joint)
RL_ARB_Blade	via J_RL_ARB_Bend	(Spherical Joint)
ground	via Slalom	(Force_Vector)
ground	via Lane_Change	(Force_Vector)
ground	via Slalom_Actual	(Force_Vector)

Part FL\_BC

Is connected to:

Chassis	via J_FL_BC_center	(Revolute Joint)
FL_Prod	via J_FL_Prod_to_BC	(Spherical Joint)

Part FL\_Upright

Is connected to:

FL_UCA	via J_FL_UCA_outer	(Spherical Joint)
FL_LCA	via J_FL_LCA_outer	(Spherical Joint)
FL_Tierod	via J_FL_Tierod_outer	(Spherical Joint)
FL_Tire	via general_motion_1.motion_t1	(Point Motion)
FL_Tire	via general_motion_1.motion_t2	(Point Motion)
FL_Tire	via general_motion_1.motion_t3	(Point Motion)
FL_Tire	via general_motion_1.motion_r1	(Point Motion)
FL_Tire	via general_motion_1.motion_r2	(Point Motion)
FL_Tire	via general_motion_1.motion_r3	(Point Motion)
FL_Tire	via J_FL_Wheel	(Revolute Joint)

```

Part FL_LCA
Is connected to:
  Chassis      via J_FL_LCA_front      (Spherical Joint)
  Chassis      via J_FL_LCA_rear      (Spherical Joint)
  FL_Upright   via J_FL_LCA_outer     (Spherical Joint)
  Chassis      via FL_Shock.sforce    (Single_Component_Force)
  FL_ARB_Droplink via J_FL_ARB_drop_link (Spherical Joint)

Part FL_UCA
Is connected to:
  FL_Prod      via J_FL_Prod_to_UCA   (Spherical Joint)
  Chassis      via J_FL_UCA_front     (Spherical Joint)
  Chassis      via J_FL_UCA_rear     (Spherical Joint)
  FL_Upright   via J_FL_UCA_outer     (Spherical Joint)

Part FL_Prod
Is connected to:
  FL_BC        via J_FL_Prod_to_BC   (Spherical Joint)
  FL_UCA       via J_FL_Prod_to_UCA   (Spherical Joint)

Part FL_Tierod
Is connected to:
  FL_Upright   via J_FL_Tierod_outer  (Spherical Joint)
  Steering_Rack via J_FL_Tierod_to_Steering (Spherical Joint)

Part FL_Tire
Is connected to:
  FL_Upright   via general_motion_1.motion_t1 (Point Motion)
  FL_Upright   via general_motion_1.motion_t2 (Point Motion)
  FL_Upright   via general_motion_1.motion_t3 (Point Motion)
  FL_Upright   via general_motion_1.motion_r1 (Point Motion)
  FL_Upright   via general_motion_1.motion_r2 (Point Motion)
  FL_Upright   via general_motion_1.motion_r3 (Point Motion)
  FL_Upright   via J_FL_Wheel         (Revolute Joint)
  ground       via CONTACT_FL        (Contact)

Part FR_BC
Is connected to:
  Chassis      via J_FR_BC_center     (Revolute Joint)
  FR_Prod      via J_FR_Prod_to_BC   (Spherical Joint)

Part FR_Upright
Is connected to:
  FR_UCA       via J_FR_UCA_outer     (Spherical Joint)
  FR_LCA       via J_FR_LCA_outer     (Spherical Joint)
  FR_Tierod    via J_FR_Tierod_outer  (Spherical Joint)
  FR_Tire      via J_FR_Wheel         (Revolute Joint)

Part FR_LCA
Is connected to:
  Chassis      via J_FR_LCA_front     (Spherical Joint)
  Chassis      via J_FR_LCA_rear     (Spherical Joint)
  FR_Upright   via J_FR_LCA_outer     (Spherical Joint)
  Chassis      via FR_Shock.sforce    (Single_Component_Force)
  FR_ARB_Droplink via J_FR_ARB_drop_link (Spherical Joint)

Part FR_UCA
Is connected to:
  FR_Prod      via J_FR_Prod_to_UCA   (Spherical Joint)
  Chassis      via J_FR_UCA_front     (Spherical Joint)
  Chassis      via J_FR_UCA_rear     (Spherical Joint)
  FR_Upright   via J_FR_UCA_outer     (Spherical Joint)

Part FR_Prod
Is connected to:
  FR_BC        via J_FR_Prod_to_BC   (Spherical Joint)
  FR_UCA       via J_FR_Prod_to_UCA   (Spherical Joint)

```

```

Part FR_Tierod
Is connected to:
  FR_Upright      via J_FR_Tierod_outer      (Spherical Joint)
  Steering_Rack   via J_FR_Tierod_to_Steering (Spherical Joint)

Part FR_Tire
Is connected to:
  FR_Upright      via J_FR_Wheel              (Revolute Joint)
  ground          via CONTACT_FR              (Contact)

Part Steering_Rack
Is connected to:
  Chassis         via J_Steering              (Translational Joint)
  FL_Tierod       via J_FL_Tierod_to_Steering (Spherical Joint)
  FR_Tierod       via J_FR_Tierod_to_Steering (Spherical Joint)

Part RL_BC
Is connected to:
  Chassis         via RL_Shock.sforce         (Single_Component_Force)
  Chassis         via J_RL_BC_center          (Revolute Joint)
  RL_Prod         via J_RL_Prod_to_BC         (Spherical Joint)

Part RL_Upright
Is connected to:
  RL_UCA          via J_RL_UCA_outer          (Spherical Joint)
  RL_LCA          via J_RL_LCA_outer          (Spherical Joint)
  RL_Tierod       via J_RL_Tierod_outer       (Spherical Joint)
  RL_Tire         via J_RL_Wheel              (Revolute Joint)

Part RL_LCA
Is connected to:
  RL_Prod         via J_RL_Prod_to_LCA        (Spherical Joint)
  Chassis         via J_RL_LCA_front          (Spherical Joint)
  Chassis         via J_RL_LCA_rear           (Spherical Joint)
  RL_Upright     via J_RL_LCA_outer          (Spherical Joint)
  RL_ARB_Droplink via J_RL_ARB_drop_link      (Spherical Joint)

Part RL_UCA
Is connected to:
  Chassis         via J_RL_UCA_front          (Spherical Joint)
  Chassis         via J_RL_UCA_rear           (Spherical Joint)
  RL_Upright     via J_RL_UCA_outer          (Spherical Joint)

Part RL_Prod
Is connected to:
  RL_BC          via J_RL_Prod_to_BC         (Spherical Joint)
  RL_LCA         via J_RL_Prod_to_LCA        (Spherical Joint)

Part RL_Tierod
Is connected to:
  RL_Upright     via J_RL_Tierod_outer       (Spherical Joint)
  Chassis        via J_RL_Tierod_inner       (Spherical Joint)

Part RL_Tire
Is connected to:
  RL_Upright     via J_RL_Wheel              (Revolute Joint)
  ground         via CONTACT_RL              (Contact)

Part RR_BC
Is connected to:
  Chassis         via RR_Shock.sforce         (Single_Component_Force)
  Chassis         via J_RR_BC_center          (Revolute Joint)
  RR_Prod         via J_RR_Prod_to_BC         (Spherical Joint)

```

```

Part RR_Upright
Is connected to:
  RR_UCA          via J_RR_UCA_outer      (Spherical Joint)
  RR_LCA          via J_RR_LCA_outer      (Spherical Joint)
  RR_Tierod       via J_RR_Tierod_outer   (Spherical Joint)
  RR_Tire         via J_RR_Wheel        (Revolute Joint)

Part RR_LCA
Is connected to:
  RR_Prod         via J_RR_Prod_to_LCA      (Spherical Joint)
  Chassis         via J_RR_LCA_front   (Spherical Joint)
  Chassis         via J_RR_LCA_rear    (Spherical Joint)
  RR_Upright      via J_RR_LCA_outer   (Spherical Joint)
  RR_ARB_Droplink via J_RR_ARB_drop_link   (Spherical Joint)

Part RR_UCA
Is connected to:
  Chassis         via J_RR_UCA_front   (Spherical Joint)
  Chassis         via J_RR_UCA_rear    (Spherical Joint)
  RR_Upright      via J_RR_UCA_outer   (Spherical Joint)

Part RR_Prod
Is connected to:
  RR_BC           via J_RR_Prod_to_BC   (Spherical Joint)
  RR_LCA          via J_RR_Prod_to_LCA   (Spherical Joint)

Part RR_Tierod
Is connected to:
  RR_Upright      via J_RR_Tierod_outer   (Spherical Joint)
  Chassis         via J_RR_Tierod_inner (Spherical Joint)

Part RR_Tire
Is connected to:
  RR_Upright      via J_RR_Wheel        (Revolute Joint)
  ground          via CONTACT_RR        (Contact)

Part FL_ARB_Droplink
Is connected to:
  FL_LCA          via J_FL_ARB_drop_link   (Spherical Joint)
  FL_ARB_Blade    via J_FL_ARB_leaf_link   (Spherical Joint)

Part FL_ARB_Blade
Is connected to:
  FR_ARB_Blade    via F_ARB.sforce         (Single_Component_Force)
  FL_ARB_Droplink via J_FL_ARB_leaf_link   (Spherical Joint)
  Chassis         via J_FL_ARB_Bend    (Spherical Joint)
  FR_ARB_Blade    via TORSION_SPRING_2.sforce (Single_Component_Force)

Part FR_ARB_Droplink
Is connected to:
  FR_LCA          via J_FR_ARB_drop_link   (Spherical Joint)
  FR_ARB_Blade    via J_FR_ARB_leaf_link   (Spherical Joint)

Part FR_ARB_Blade
Is connected to:
  FL_ARB_Blade    via F_ARB.sforce         (Single_Component_Force)
  FR_ARB_Droplink via J_FR_ARB_leaf_link   (Spherical Joint)
  Chassis         via J_FR_ARB_Bend    (Spherical Joint)
  FL_ARB_Blade    via TORSION_SPRING_2.sforce (Single_Component_Force)

Part RL_ARB_Droplink
Is connected to:
  RL_LCA          via J_RL_ARB_drop_link   (Spherical Joint)
  RL_ARB_Blade    via J_RL_ARB_leaf_link   (Spherical Joint)

```



```

Part RL_ARB_Blade
Is connected to:
  RR_ARB_Blade      via  R_ARB.sforce      (Single_Component_Force)
  RL_ARB_Droplink  via  J_RL_ARB_leaf_link  (Spherical Joint)
  Chassis           via  J_RL_ARB_Bend    (Spherical Joint)

Part RR_ARB_Droplink
Is connected to:
  RR_LCA           via  J_RR_ARB_drop_link  (Spherical Joint)
  RR_ARB_Blade     via  J_RR_ARB_leaf_link  (Spherical Joint)

Part RR_ARB_Blade
Is connected to:
  RL_ARB_Blade     via  R_ARB.sforce      (Single_Component_Force)
  RR_ARB_Droplink  via  J_RR_ARB_leaf_link  (Spherical Joint)
  Chassis          via  J_RR_ARB_Bend    (Spherical Joint)

```

## Appendix D. ADAMS 2018 FSAE MODEL STRUCTURE (By Connections)

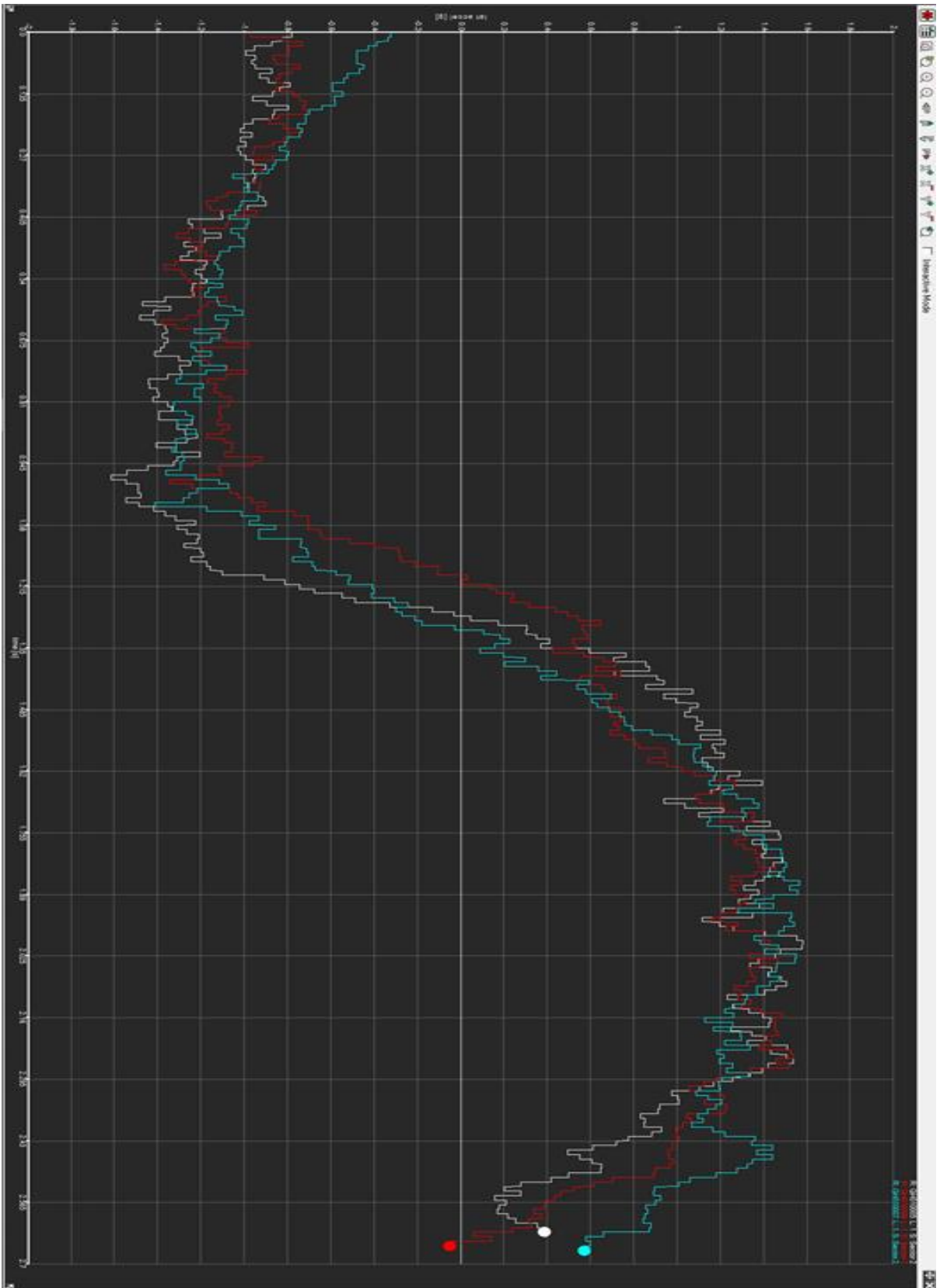
Topology of model: UNM\_fsae\_2018

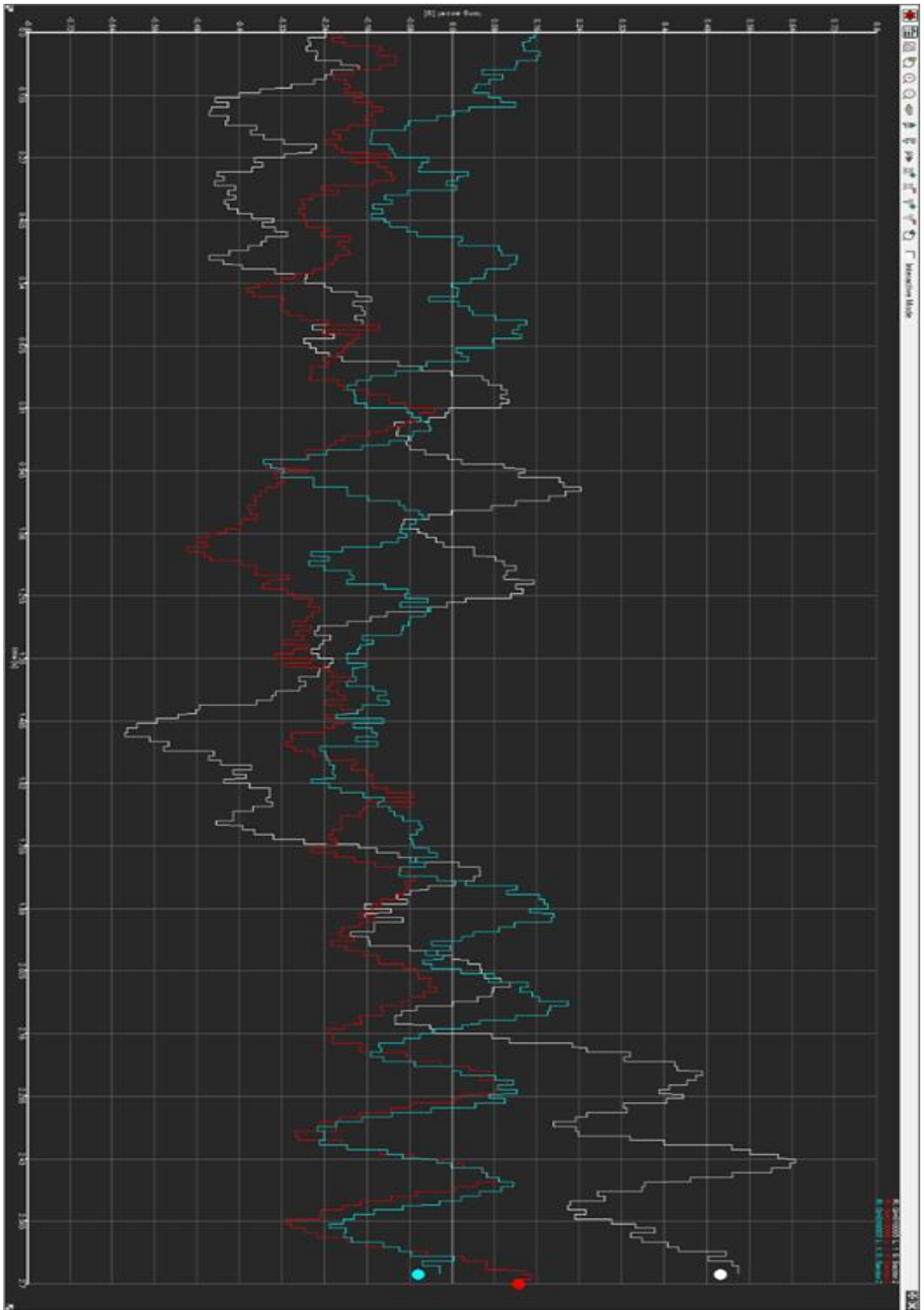
Ground Part: ground

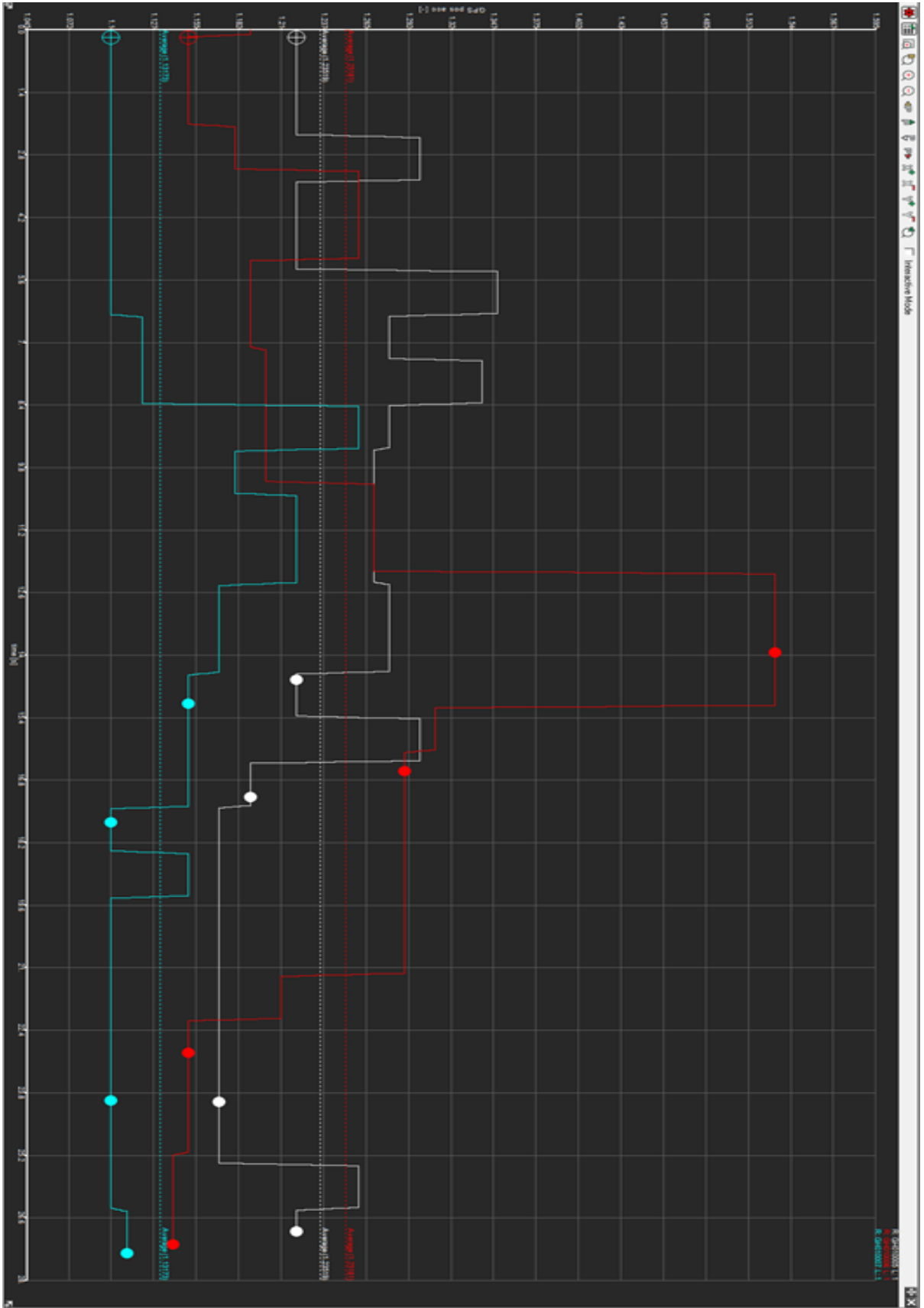
CONTACT_FL	connects	FL_Tire	with	ground	(Contact)
CONTACT_FR	connects	FR_Tire	with	ground	(Contact)
CONTACT_RL	connects	RL_Tire	with	ground	(Contact)
CONTACT_RR	connects	RR_Tire	with	ground	(Contact)
J_FL_BC_center	connects	FL_BC	with	Chassis	(Revolute Joint)
J_FL_Prod_to_BC	connects	FL_Prod	with	FL_BC	(Spherical Joint)
J_FL_Prod_to_UCA	connects	FL_Prod	with	FL_UCA	(Spherical Joint)
J_FL_UCA_front	connects	FL_UCA	with	Chassis	(Spherical Joint)
J_FL_UCA_rear	connects	FL_UCA	with	Chassis	(Spherical Joint)
J_FL_UCA_outer	connects	FL_UCA	with	FL_Upright	(Spherical Joint)
J_FL_LCA_front	connects	FL_LCA	with	Chassis	(Spherical Joint)
J_FL_LCA_rear	connects	FL_LCA	with	Chassis	(Spherical Joint)
J_FL_LCA_outer	connects	FL_LCA	with	FL_Upright	(Spherical Joint)
J_FL_Tierod_outer	connects	FL_Tierod	with	FL_Upright	(Spherical Joint)
J_FL_Wheel	connects	FL_Tire	with	FL_Upright	(Revolute Joint)
J_FR_BC_center	connects	FR_BC	with	Chassis	(Revolute Joint)
J_FR_Prod_to_BC	connects	FR_Prod	with	FR_BC	(Spherical Joint)
J_FR_Prod_to_UCA	connects	FR_Prod	with	FR_UCA	(Spherical Joint)
J_FR_UCA_front	connects	FR_UCA	with	Chassis	(Spherical Joint)
J_FR_UCA_rear	connects	FR_UCA	with	Chassis	(Spherical Joint)
J_FR_UCA_outer	connects	FR_UCA	with	FR_Upright	(Spherical Joint)
J_FR_LCA_front	connects	FR_LCA	with	Chassis	(Spherical Joint)
J_FR_LCA_rear	connects	FR_LCA	with	Chassis	(Spherical Joint)
J_FR_LCA_outer	connects	FR_LCA	with	FR_Upright	(Spherical Joint)
J_FR_Tierod_outer	connects	FR_Tierod	with	FR_Upright	(Spherical Joint)
J_FR_Wheel	connects	FR_Tire	with	FR_Upright	(Revolute Joint)
J_FL_Tierod_to_Steering	connects	FL_Tierod	with	Steering_Rack	(Spherical Joint)
J_FR_Tierod_to_Steering	connects	FR_Tierod	with	Steering_Rack	(Spherical Joint)
J_Steering	connects	Steering_Rack	with	Chassis	(Translational Joint)
J_RL_BC_center	connects	RL_BC	with	Chassis	(Revolute Joint)
J_RL_Prod_to_BC	connects	RL_Prod	with	RL_BC	(Spherical Joint)
J_RL_Prod_to_LCA	connects	RL_Prod	with	RL_LCA	(Spherical Joint)
J_RL_UCA_front	connects	RL_UCA	with	Chassis	(Spherical Joint)
J_RL_UCA_rear	connects	RL_UCA	with	Chassis	(Spherical Joint)
J_RL_UCA_outer	connects	RL_UCA	with	RL_Upright	(Spherical Joint)
J_RL_LCA_front	connects	RL_LCA	with	Chassis	(Spherical Joint)
J_RL_LCA_rear	connects	RL_LCA	with	Chassis	(Spherical Joint)
J_RL_LCA_outer	connects	RL_LCA	with	RL_Upright	(Spherical Joint)
J_RL_Tierod_outer	connects	RL_Tierod	with	RL_Upright	(Spherical Joint)
J_RL_Wheel	connects	RL_Tire	with	RL_Upright	(Revolute Joint)
J_RR_BC_center	connects	RR_BC	with	Chassis	(Revolute Joint)
J_RR_Prod_to_BC	connects	RR_Prod	with	RR_BC	(Spherical Joint)
J_RR_Prod_to_LCA	connects	RR_Prod	with	RR_LCA	(Spherical Joint)
J_RR_UCA_front	connects	RR_UCA	with	Chassis	(Spherical Joint)
J_RR_UCA_rear	connects	RR_UCA	with	Chassis	(Spherical Joint)
J_RR_UCA_outer	connects	RR_UCA	with	RR_Upright	(Spherical Joint)
J_RR_LCA_front	connects	RR_LCA	with	Chassis	(Spherical Joint)
J_RR_LCA_rear	connects	RR_LCA	with	Chassis	(Spherical Joint)
J_RR_LCA_outer	connects	RR_LCA	with	RR_Upright	(Spherical Joint)
J_RR_Tierod_outer	connects	RR_Tierod	with	RR_Upright	(Spherical Joint)
J_RR_Wheel	connects	RR_Tire	with	RR_Upright	(Revolute Joint)
J_RL_Tierod_inner	connects	RL_Tierod	with	Chassis	(Spherical Joint)
J_RR_Tierod_inner	connects	RR_Tierod	with	Chassis	(Spherical Joint)
J_FL_ARB_drop_link	connects	FL_LCA	with	FL_ARB_Droplink	(Spherical Joint)
J_FL_ARB_leaf_link	connects	FL_ARB_Droplink	with	FL_ARB_Blade	(Spherical Joint)
J_FL_ARB_Bend	connects	FL_ARB_Blade	with	Chassis	(Spherical Joint)
J_FR_ARB_drop_link	connects	FR_LCA	with	FR_ARB_Droplink	(Spherical Joint)
J_FR_ARB_leaf_link	connects	FR_ARB_Droplink	with	FR_ARB_Blade	(Spherical Joint)
J_FR_ARB_Bend	connects	FR_ARB_Blade	with	Chassis	(Spherical Joint)
J_RL_ARB_drop_link	connects	RL_LCA	with	RL_ARB_Droplink	(Spherical Joint)
J_RL_ARB_leaf_link	connects	RL_ARB_Droplink	with	RL_ARB_Blade	(Spherical Joint)
J_RL_ARB_Bend	connects	RL_ARB_Blade	with	Chassis	(Spherical Joint)
J_RR_ARB_drop_link	connects	RR_LCA	with	RR_ARB_Droplink	(Spherical Joint)
J_RR_ARB_leaf_link	connects	RR_ARB_Droplink	with	RR_ARB_Blade	(Spherical Joint)
J_RR_ARB_Bend	connects	RR_ARB_Blade	with	Chassis	(Spherical Joint)
FL_Shock.sforce	connects	Chassis	with	FL_LCA	(Single_Component_Force)
FR_Shock.sforce	connects	Chassis	with	FR_LCA	(Single_Component_Force)
RL_Shock.sforce	connects	Chassis	with	RL_BC	(Single_Component_Force)
RR_Shock.sforce	connects	Chassis	with	RR_BC	(Single_Component_Force)

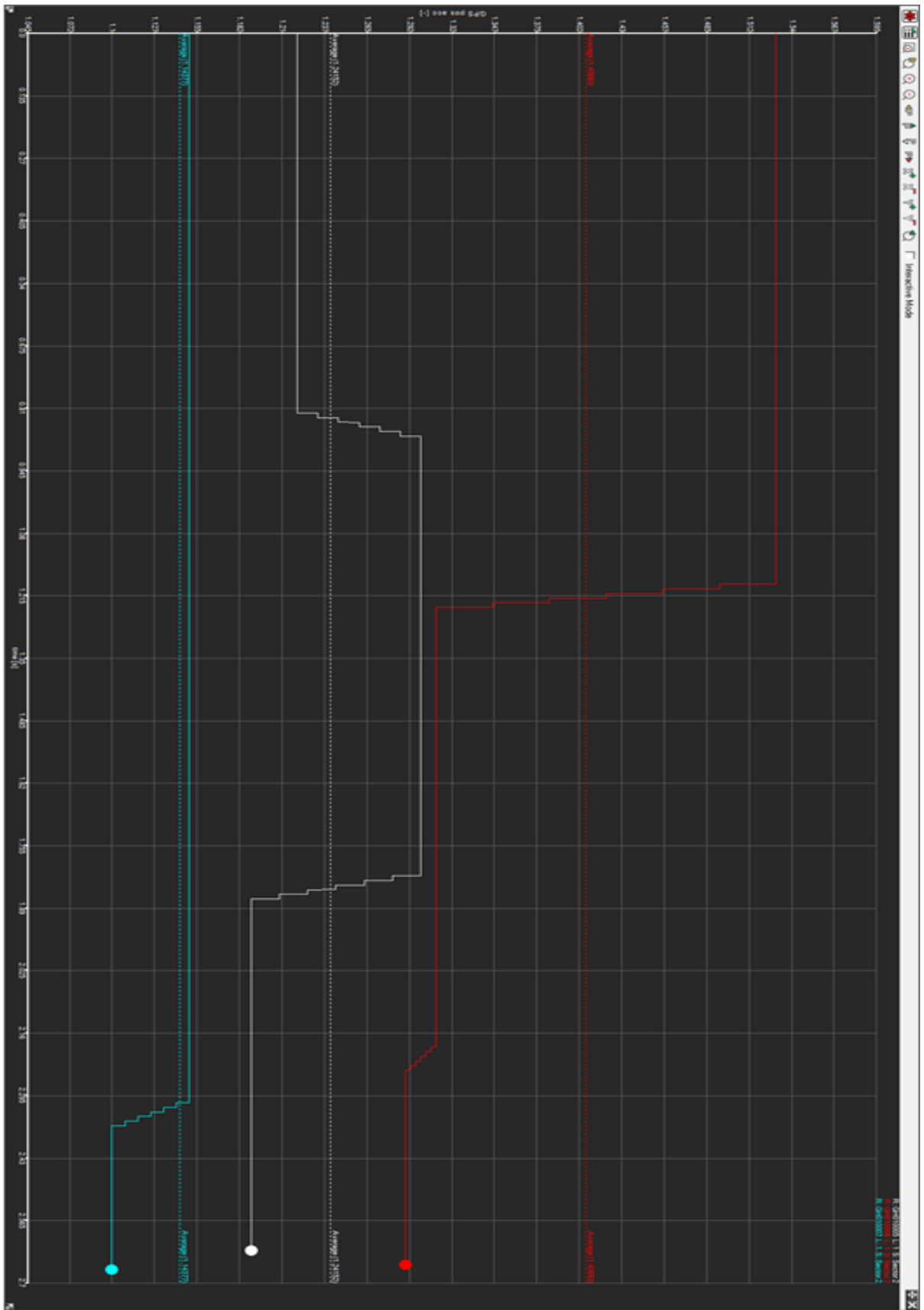
F_ARB.sforce	connects	FL_ARB_Blade	with	FR_ARB_Blade	(Single_Component_Force)
R_ARB.sforce	connects	RL_ARB_Blade	with	RR_ARB_Blade	(Single_Component_Force)
general_motion_1.motion_t1	connects	FL_Tire	with	FL_Upright	(Point Motion)
general_motion_1.motion_t2	connects	FL_Tire	with	FL_Upright	(Point Motion)
general_motion_1.motion_t3	connects	FL_Tire	with	FL_Upright	(Point Motion)
general_motion_1.motion_r1	connects	FL_Tire	with	FL_Upright	(Point Motion)
general_motion_1.motion_r2	connects	FL_Tire	with	FL_Upright	(Point Motion)
general_motion_1.motion_r3	connects	FL_Tire	with	FL_Upright	(Point Motion)
Slalom	connects	Chassis	with	ground	(Force_Vector)
Lane_Change	connects	Chassis	with	ground	(Force_Vector)
TORSION_SPRING_2.sforce	connects	FL_ARB_Blade	with	FR_ARB_Blade	(Single_Component_Force)
Slalom_Actual	connects	Chassis	with	ground	(Force_Vector)

# Appendix E. Race Technology/ GoPro Data









## References

Berretta, J. and da Silva, G., "FSAE suspension development in virtual environment," SAE Technical Paper 2018-36-0231, 2018.

D. Seward. Race Car Design. Palgrave Macmillan, 2015. ISBN: 978-1560915263.

Luque, P., Mántaras, D. A., & Pello, A. (2012). Racing car chassis optimization using the finite element method, multi-body dynamic simulation and data acquisition. *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*, 227(1), 3-11. doi:10.1177/1754337112444517

Marzuki, M. A., Bakar, M. A., & Azmi, M. F. (2015). Designing Space Frame Race Car Chassis Structure Using Natural Frequencies Data From Ansys Mode Shape Analysis. *International Journal of Information Systems and Engineering*, 3(1), 54-63. doi:10.24924/ijise/2015.11/v3.iss1/54.63

Student Editions. *mscsoftware*. Retrieved from <https://www.mscsoftware.com/student-editions>

The Editors of Encyclopaedia Britannica. (2019, June 17). Automobile racing. Retrieved from <https://www.britannica.com/sports/automobile-racing>.

Yang, L., Li, Q., Wang, C., and Zhang, Y., "Loads Analysis and Optimization of FSAE Race Car Frame," SAE Technical Paper 2017-01-0423, 2017, doi:10.4271/2017-01-0423.