Formulation of a Path-Following Joint for Multibody System Dynamics

by

Andrew Hall

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2015

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The development and validation of a new multibody joint that constrains a body to follow a spatial path and an orientation defined by a user is presented. The resulting joint has a single degree of freedom (DOF), and maintains equivalent kinematic behaviour when compared to higher-fidelity models. As such, it is referred to as a single-DOF equivalent kinematic (SEK) joint. The primary application of this joint is in the reduction of complex multibody systems, specifically vehicle suspensions.

The first formulation of the joint is developed using the user interface of MapleSim. Starting with a planar particle joint, the theory is extended to a full 3D rigid body constraint. At each development stage, the joint is successfully validated against conventional models in both Adams and MapleSim. This formulation of the joint results in the kinematic pair being represented by a system of differential algebraic equations (DAEs) which is not the desired functionality, and so a second formulation is developed. By removing the constraint of using the MapleSim user interface, the formulation can be developed from first principles. Using the path-length as the coordinate for the joint, and the Frenet-Serret equations to compute the motion and reaction spaces, the kinematic pair can be represented by a single ordinary differential equation (ODE). The theory is implemented in the MapleSim source code using the symbolic computing language Maple.

The theory of the SEK joint can be extended to create different joints. The first is the compliant SEK joint. In this version of the joint, the body is constrained to move along a spatial path using a simple linear bushing model. The compliant SEK joint is useful for modeling the suspension systems of passenger cars as bushings are used extensively in these systems to increase passenger comfort. The second extension is to add an additional DOF to the SEK joint to created the double-DOF equivalent kinematic (DEK) joint. The DEK joint is useful for modelling steered suspension systems as the steering introduces an additional DOF to the suspension. The envelope of motion for the steered wheel is a surface rather than a spatial line.

Once the joints are successfully validated, three example applications of the joint are shown. In the first, rigid, compliant and steered suspension models are developed and compared against high-fidelity models in Adams/Car and MapleSim. Next, a full vehicle model is assembled using the suspension models and compared against an equivalent high-fidelity full vehicle model built in MapleSim. The comparisons show the accuracy of the SEK joint as well as the simulation speed improvements it can offer compared with conventional modelling techniques. The second example, from the domain of biomechanics, shows a knee model created using the SEK joint. Finally, a roller coaster model is created to demonstrate the flexibility of the path generation algorithm to create splines that represent complex paths.

Acknowledgements

I would like to thank ...

my supervisor, Professor John McPhee, for his unwavering support since I arrived at University of Waterloo. By both teaching and providing the environment to learn independently you have allowed me to grow immensely, both personally and professionally.

Dr. Chad Schmitke, for the countless hours that he spent helping me with the MapleSim implementation aspects of this work.

the members of my committee: Professor Nasser Azad, Professor Steve Lambert and Dr. Chad Schmitke from the University of Waterloo, and Professor Subash Rakheja from Concordia University for taking the time to read my thesis and participate in my final defense.

all past and present members of the Motion Research Group (MoRG) for providing a friendly and intellectually stimulating environment within our lab.

the many friends I have made in the Waterloo area since arriving here.

and, last but not least, my parents, Jim and Rowena, and sisters, Laura and Madeleine, for their support and encouragement.

Dedication

Ellie

Table of Contents

List of Tables

List of Figures

Sign Conventions

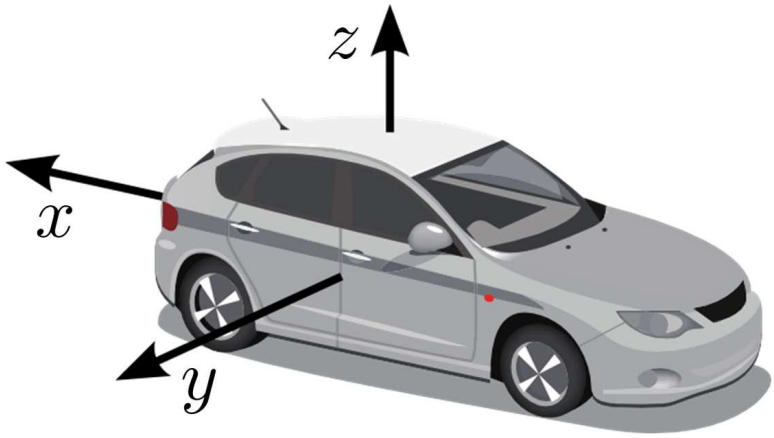


Figure 1: Vehicle sign convention [1]

Notes

All simulation results presented in this thesis have been obtained using a computer with the following specifications:

- Dell Optiplex 780
- Intel Core2 Duo E7500
- 4.0 GB RAM
- Windows 7 Professional 64-bit

Adams/View 2013.2 and Adams/Car 2013.2 were used to obtain all Adams results

MapleSim 6.4 and Maple 18 were used to obtain all MapleSim results

Chapter 1

Introduction

Vehicle dynamics is the study of the motion of a vehicle, and the forces causing this motion. Computer simulation is a preferred tool over real world testing of the physical system as it offers an opportunity for significant savings of both time and money. Furthermore, test cases that may be too dangerous or difficult to carry out in the real world can often be easily simulated in a virtual setting. To best understand the dynamics of a vehicle, a multibody model is used. The development of accurate multibody vehicle dynamics models can be a time consuming process, and so alternate approaches are often sought. Ideally, the alternate approach will offer a significant reduction in the development time without sacrificing model accuracy or scope. The development of real-time capable vehicle models is another popular research topic. Due to the physical requirements of a vehicle's suspension, they generally consist of closed kinematic loops and bushings that require large sets of equations to represent their physics, leading to slow simulation times. In this case the goal is to offer reduced simulation time, while maintaining an acceptable degree of accuracy in the model.

In this thesis, a path-following joint is developed with the goal of using it to create reduced suspension models. Aside from the advantage of quicker simulation times, the joint provides a way for the user to quickly and easily define a vehicle's suspension. To generate a high-fidelity model, extensive geometric information is required that can be costly and time-consuming to obtain. Furthermore, the performance implications of varying the suspension curves (such as the camber variation as a result of the vertical displacement of the wheel hub) can be tested directly without the need to tune specific suspension geometry properties to obtain the desired kinematic response. This path-following joint can be used to represent

any kind of suspension and is completely independent of the geometric topology of the system.

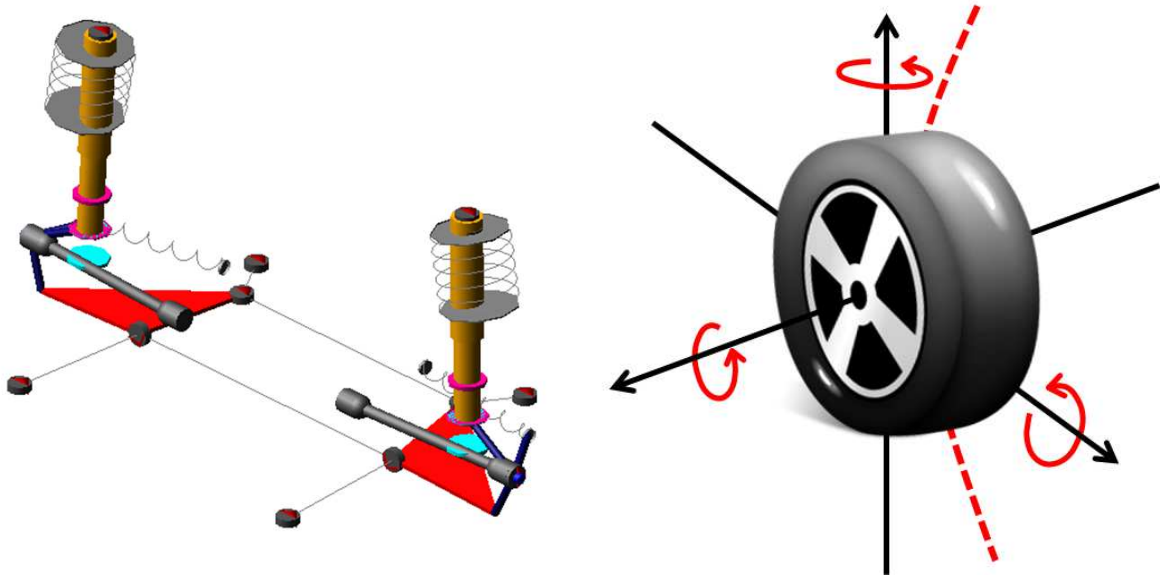
The joint is realised as a path-following constraint in which a body is constrained to follow a spatial path and orientation, defined by the user. Path-following joints also have uses in other fields beyond vehicle dynamics. One example of this is in biomechanics where path-following joints might be used to represent complex biological joints, such as the knee. The path-following joint has 1-degree of freedom (DOF) and can be used to accurately represent the kinematics of complex 1-DOF systems in a simple fashion; as such it is called the single-DOF equivalent kinematic (SEK) joint. The SEK joint is also extended to create a 2-DOF equivalent kinematic (DEK) joint. In this joint, a body is constrained to a surface, rather than a spatial path.

1.1 Background

Since the modern automobile was invented at the beginning of the 1900s, engineers have been working to understand the dynamics of these systems in an attempt to improve performance, comfort, and safety. Before computer simulation became the leading tool for development in the 1980s [2], the pioneers of vehicle dynamics research developed their own physical test rigs and laid the ground work that all vehicle dynamicists depend on today. Some of their theories still persist in modern day vehicle dynamics research; for example, Olley's theory that front axle pitch stiffness be softer than that of the rear (referred to as Olley's tuning) [3] is still used today, and has been shown to be a good solution, especially for high speed stability [4–6].

Deriving and solving the equations of motion of a multibody system (MBS) by hand is not a reasonable pursuit except in the most trivial of cases. As digital computers became more commonplace, numerous multibody simulation programs appeared, with MSC Software's Adams[1] becoming one of the most popular. In 1977, Orlandea et al. published a paper describing the use of Adams to simulate a vehicle suspension [7], bringing together multibody simulation and vehicle dynamics research. It is now commonplace for vehicle dynamicists to study all aspects of a vehicle's dynamical behaviour using multibody simulation software [8].

[1] Adams, Adams/Car and Adams/View are registered trademarks of MSC Software Corp.



(a) Geometric suspension model

(b) Conceptual visualization of characteristic suspension model

Figure 1.1: Different modelling techniques for suspension systems

1.2 Motivation

According to Cao et al. [9], there are three fundamental factors of a suspension design that affect its performance: the spring, the damper, and suspension kinematics and compliance (K&C). The spring and damper have been extensively studied using a variety of models, from simple linear quarter-car models to highly non-linear multibody models. Unfortunately, the K&C is neglected in many models due to the need for a more complex multibody model when compared to the simple models that can be used for spring and damper studies.

A traditional approach to creating a vehicle suspension model is to create a "geometric" model in which the physical topology of the system is reproduced accurately in a multibody model. In this case, each component in the system is included in the model. This approach certainly seems ideal in terms of accuracy; however these models can be time-consuming to construct as many parameters are required. An example geometric model created in Adams/Car is shown in Figure 1.1a. An alternative approach is to use a "characteristic" modeling approach in which the K&C characteristics of the suspension are enforced by

some type of multibody constraint. Development of this type of suspension model can be much less time-consuming, but care must be taken when defining the multibody constraint that the accuracy of the model is not unduly compromised. By generating the model in this way, independent of the physical structure of the system, a single model can be used to represent any type of suspension. The concept of a "characteristic" suspension model can be seen in Figure 1.1b. It is for such "characteristic" suspension models that the developed SEK joint should be used.

Since DaimlerChrysler introduced anti-lock braking systems to the consumer vehicle market in 1979 [10], vehicle dynamic control systems have become commonplace in road vehicles. An ideal way to develop these control systems is using hardware-in-the-loop (HIL) testing. It wasn't until the late 1980s that the first test rigs were built for electronic control unit (ECU) development [11]. To develop a controller and test it using an HIL setup, a model is required that is capable of running in real-time. For a simulation to be considered as running in real-time, the simulation must occur in less time than the real world event. It is possible to build a better controller if more model detail offering improved accuracy, such as detailed kinematic and compliance (K&C) information, can be included in a model while maintaining real-time capability.

1.3 Challenges

The first challenge in this work is to formulate the joint in a way that gives the simplest mathematical description of the constraint, but also makes it easy for a user to define the parameters of the joint. This will allow the joint to meet the two goals of fast model development and fast simulation times.

Developing a procedure to develop mathematical functions that define a generic spatial path is a second challenge presented by this work. The path generation procedure should be able to take data and automatically generate a spatial path in a way that complements the chosen parameterization used in the joint formulation. The goal is to be able to generate a spatial path directly from experimental data, whether it be obtained from a physical test or a simulation using a higher-fidelity model.

The final challenge is the implementation of the joint in MapleSim. The symbolic simplification and optimization algorithms used by MapleSim to build the system equations

generate dynamic simulation code that is computationally efficient. Coupling the conceptual simplicity of the SEK and DEK joints with the optimized simulation code generated by MapleSim will allow models built with the SEK and DEK joints to outperform more traditional models. The implementation must ensure the joint is not only kinematically correct, but also accurately represents the dynamics of the system. To ensure accurate vehicle performance it is necessary to ensure that the suspension model is capable of transmitting the correct reaction forces generated by the tyre to the chassis.

1.4 Applications

In the field of vehicle dynamics, the SEK joint has numerous applications. The simpler parameterization when compared to the geometric parameterization used in programs such as Adams/Car, means that users will be able to quickly generate suspension models for a wide variety of offline studies. Because of its real-time performance, the joint could be used in HIL setups and model-based control systems. The joint also has potential applications other than vehicle dynamics—in fields such as rail vehicle dynamics to model the track, and in biomechanics where the standard set of multibody joint constraints cannot be used to accurately represent complex biological joints [12, 13].

1.5 Document organization

Chapter 2 is a review of literature relevant to this research; Section 2.1 discusses some common multibody vehicle dynamics software and modelling techniques. Next, in Section 2.2 the current research into path-following joints is presented, as this is fundamentally relevant to the theory in this thesis. In Section 2.4 some information on interpolation and spline fitting is presented. Splines play an integral role in the definition of the kinematics enforced by the SEK joint. In Chapter 3 the theory of the path following SEK joint is presented, along with a two-DOF extension of the work: the double-DOF equivalent kinematic (DEK) joint. In Section 3.1 the initial formulation of the SEK joint is shown. This formulation did not result in the desired level of mathematical simplicity (namely, only ODEs), but it formed the basis for the reformulation of the SEK joint that is presented in Section 3.2. In this section, the final formulation of the SEK joint is presented along

with the algorithms that are used to generate the spatial path used in the SEK joint. In Section 3.3 the SEK joint theory is extended for use in the DEK joint. In Chapter 4 some practical applications of the SEK and DEK joints are shown. Section 4.1 deals with vehicle dynamic models. Rigid, compliant and steered suspension system models are created, followed by a full vehicle model using multiple SEK and DEK joints. Section 4.2 shows a simple biomechanical knee example of the SEK joint, and in Section 4.3 a roller coaster is simulated to demonstrate some aspects of the path generation algorithm from Section 3.2. Finally, the thesis is concluded in Chapter 5 and the contributions of this research are summarized along with some recommendations for future work.

Chapter 2

Literature review

2.1 Multibody dynamic modelling

2.1.1 Adams

Adams is the most widely used multibody simulation package [14], developed by MSC Software. The general multibody modelling software is called Adams/View, but an add-on called Adams/Car is specifically geared towards vehicle dynamics simulations. Adams/Car is essentially the standard Adams/View software with a different user interface and macros that make it simpler to create vehicle models using predefined templates for standard vehicle components and suspension configurations, and run both open- and closed-loop vehicle dynamics simulations such as a lane change, or rig tests such as a four post test [1]. The method that Adams uses to assemble the system equations results in a large set of differential algebraic equations (DAEs) meaning that all but the most trivial models are not suited to real-time applications.

2.1.2 CarSim

CarSim is commercially available vehicle dynamics simulation software, based on AUTOSIM, which was initially developed by Sayers during his PhD. research work [15]. AUTOSIM was developed with the goal of automatically developing efficient, real-time capable simulation code symbolically from a multibody model definition. Before CarSim was available as a means to easily develop real-time vehicle models, researchers had been forced

to derive and fine-tune the system equations by hand, which can be time consuming and error prone [16].

In [17] Sayer presents the original vehicle model used in CarSim. Some details have changed in more recent editions of the software to offer a more accurate model. However, details of the original model are presented here as the assumptions made are still of interest in the context of current vehicle dynamics research, and the original model definition is still accessible and commonly used.

In CarSim the suspension links are not modelled, and lookup tables or coefficients are used to define the kinematic properties of a suspension system, such as camber and toe variation, as well as roll center height [18]. This departure from the geometric model definition used in programs such as Adams/Car results in a lower-fidelity model but, as Sayer points out, does make the creation of a model simpler. To develop an Adams/Car model, detailed geometric data of the vehicle is required, such as the suspension pick up points and detailed specifications for components such as bushings. This information can be difficult to obtain.

The CarSim model uses a single rigid body as the chassis with six DOF, and each of the four unsprung mass components (the wheel carrier) adds an additional translational DOF. The orientation of the unsprung mass is partially defined using a lookup table for the toe and camber as a function of wheel carrier vertical displacement, but in the original model this does not seem to be included in the dynamical equations for the wheel carrier and is only used as an input to the tyre model. The translational kinematics of the wheel carrier play an important role in the transmission of forces between the tyre contact patch and the chassis. In traditional hand-derived equations such as those presented in [2, 19], these force transmissions have been conceptualized into terms such as anti-roll, anti-pitch, anti-dive, anti-squat, and jacking [20, 21]. In the original CarSim model, the assumption is made that these translations are linear functions of vertical displacement. The user can specify a constant wheel carrier longitudinal translation coefficient as a function of wheel carrier vertical translation. The roll center height is specified as a constant distance from the ground, and the lateral wheel center translation is determined by the ratio between the roll center height and the half-track width.

Springs, dampers, bump stops, and anti-roll bars are also simplified in the CarSim

model [17]. The motion ratio, MR, of such a force component is defined as:

$$MR = \Delta C / \Delta W \tag{2.1}$$

where ΔC is the vertical displacement of the force component and ΔW is the vertical displacement of the wheel carrier. The rate, or stiffness, of the component is defined in a lookup table with the appropriate dependent variable (vertical displacement or velocity). Using the motion ratio, the effective rate of the component at the tyre contact patch can be determined by:

$$K_W = MR^2 * K_C \tag{2.2}$$

where K_W is the rate of the component at the wheel carrier and K_C is the actual rate of the component. In such an application the user does not define where on the sprung mass or the wheel carrier the force components connect. This further simplifies the model creation process.

CarSim allows the user to specify eight compliance coefficients for each wheel [22]. These handle changes in steer and camber angles, and lateral and longitudinal translations of the wheel carrier due to forces along the x- and y-axes and moments about the z-axis.

In recent releases of CarSim the kinematics definition of the suspension has been updated to implement a true multibody suspension joint defined solely by the kinematics of the suspension [22]. The lateral and longitudinal translations of the wheel carrier are defined by two lookup tables, and the orientation is determined using three lookup tables. The toe and camber angles are specified as functions of vertical displacement. The third angle is the dive angle, which is the rotation of the wheel carrier about the wheel spin axis. This allows a suspension model to be fully defined using data from a K&C test rig, or from a higher fidelity model.

This technique is not unique to CarSim, and has been described in the literature by different authors [23, 24]. The implementation by Tobolar in [23] is called the virtual axle. The model is validated by comparing a full vehicle model with a virtual axle front suspension to a reference model using a high-fidelity MacPherson strut front suspension in SIMPACK[1]. Doing so shows that the response of the virtual axle to a step steer input is an accurate representation of the high-fidelity model. The virtual axle is real-time capable and Tobolar claims a simulation time improvement of 30% compared to the reference model.

[1]SIMPACK is a trademark of SIMPACK AG.

2.1.3 MapleSim

MapleSim/Multibody is a symbolic, graph-theoretic, multi-domain simulation tool with general multibody functionality, formerly packaged as Dynaflex Pro. MapleSim is based on the symbolic programming environment of Maple. Graph theory was first developed by Leonhard Euler in 1736 [25], and has been widely applied to numerous problems in engineering [26], including multibody dynamics, as described by McPhee in [27]. An advantage of a graph theoretic approach to multibody systems is that the generation of the governing equations of motion can be derived from the system description in a systematic way. This means that a generalized algorithm can be developed to automatically generate the equations of motion for the system. Another advantage of this approach is that the coordinates of the system can be chosen in a way that gives a resulting set of equations in the form of ODEs as long as no closed kinematic loops are present [27]. The conventional approach of a numeric formulation and absolute coordinates will always result in a system of DAEs for anything but the most trivial systems.

As well as multibody dynamics, linear graph theory can also be applied to other domains. This means that in MapleSim a model can be generated that spans numerous domains, such as chemical, hydraulic and electrical, the latter becoming vital given the increasing popularity of hybrid electric and electric vehicles (HEV/EV). In [28] an HEV model is created in MapleSim with a chemical battery model, electric generator and drive motors and a multibody vehicle model.

Because MapleSim generates symbolic models it is easy to export the models to C/C++ code that can be used in environments other than MapleSim. Furthermore the system equations can be extracted and viewed by the user which can facilitate further investigation of the system, for example by carrying out sensitivity studies using symbolic differentiation [29]. While CarSim also employs symbolic code generation, it is not implemented in a multi-domain generalized modelling environment and is therefore not as versatile as MapleSim [30]; CarSim models are limited to the mechanical domain, and there are limitations on the development tools available to the user that are not present in MapleSim.

Currently there are no specific libraries implemented in MapleSim for vehicle dynamics (like Adams/Car), and so users must develop their own models using the basic building tools; see [31] for an example of how this is currently done. There is work being done to improve this and develop an equivalent of Adams/Car in MapleSim [32], but a reduced,

real-time capable suspension model is still not available. This is the focus of this work.

2.1.4 SIMPACK

In [33–35] the reduced real-time capable vehicle dynamics suspension model implemented in SIMPACK is presented, using the so-called "macro joint approach". An alternative approach to CarSim is taken in which the original parameterization of the suspension model, in terms of its geometric definition, is kept. The advantage of this approach is that the definition of the model is directly related to the physical system, making it straight forward to understand the impacts of design changes.

The principle behind this approach is to neglect the flexibility and mass properties of all suspension links. This is a reasonable assumption in a vehicle model developed with the intention of studying the handling characteristics, as only frequencies below 25Hz need to be considered [33]. This drastically reduces the number of equations and removes the closed kinematic loops, enabling the system to be described by a set of ODEs. Furthermore, all compliant joints (representing bushings) are replaced with idealized joints. This results in a kinematic suspension model, with no compliance.

Two different methods have been developed to include compliance in the macro joint approach. The first is used if the individual stiffness properties of each bushing component are known. In this case the compliance of each suspension member is defined, for example "the longitudinal movement of suspension rod A due to compliance in bushing B" [33]. The second option for defining the suspension compliance is used when the overall compliance of the wheel carrier is known [23, 36], as measured on a K&C test rig, for example. In this case a single compliance object is introduced within the wheel carrier, between the kinematical part of the wheel carrier that is attached to the chassis and the part of the wheel carrier to which the wheel is attached.

2.1.5 Other common modelling techniques

Along with those previously discussed there are also several other approaches to modelling the dynamics of vehicles that can be found in the literature, but that haven't achieved widespread usage commercially.

Understanding vehicle rollover performance has become an important topic in vehicle dynamics research for both improved mechanical design and the design and implementation of rollover prevention control systems [37]. In [38] Shim et al. review the roll-over prediction performance of a hand-derived 14-DOF [39–42] vehicle model and a fixed roll center CarSim model against a high fidelity Adams/Car model. The 14-DOF model has a chassis with 6-DOF, and 2-DOF at each of the four wheels: one for vertical suspension travel and one for wheel spin. The kinematics of the suspension is simplified, with only camber angle varying as the wheel travels vertically, and a Pacejka tyre model is used. The response to a J-turn input of both the 14-DOF and CarSim models match that of the Adams/Car model reasonably well until the vehicle starts to roll aggressively and wheel lift off occurs. Shim et al. propose that the differences exhibited by the models before wheel lift off occurs is likely due to the different handling of the roll centers in each of the models. In the 14-DOF model, the roll center is located at a fixed distance from the sprung mass CG and in CarSim the roll center is located at a fixed distance from the ground. Also, the roll center will migrate laterally as the vehicle rolls. This will result in a variation of both spring and roll stiffness characteristics of the vehicle. In Adams/Car this is not an issue as the model is a high-fidelity representation of the vehicle's suspension geometry. Furthermore in the 14-DOF model the compliance characteristics of the suspension are not considered.

Other vehicle models such as the 8-DOF [43–45] full vehicle model used for roll studies, the 7-DOF model [46] used for ride studies, the various half-car models used to study pitch and roll characteristics, and the quarter-car model used extensively in ride simulations and active suspension design [47, 48] will not be discussed here. Rauh [49] provides a listing of various modelling techniques for vehicle and component models, and Cao et al. [9] provide an extensive review of current vehicle dynamics modelling techniques.

2.1.6 Model development and validation

Physical rig testing of a vehicle is often undertaken to obtain experimental data that can be used to ensure a mathematical model is an accurate representation of the vehicle under study. A common starting point in this process is a Kinematics and Compliance (K&C) test. This is a full vehicle rig test in which the vehicle is bolted to a large table and forces are applied to the tyre contact patches along the axes that allow movement, from which the kinematics can be measured, and along axes that constrain movement, from which the compliance can be measured [50, 51]. To develop an accurate high-fidelity geometric

suspension model that matches the K&C test data, a correlation procedure must be carried out, similar to that in [52, 53]. In both papers a geometric model is fit to experimental K&C data by tuning the parameters of the geometric model. In general, to obtain a good correlation to the kinematics test results, the model hard point locations are adjusted. For the compliance correlation, the bushing stiffnesses can be adjusted. Rao et al. [52] carried out this process manually after running designed experiments to create response surfaces to understand the sensitivities of the suspension behaviour to the individual hard point locations and bushing stiffnesses. Alternatively, Hall and McPhee [53] developed a software framework so that MATLAB optimization routines could be used to identify the optimal parameters for their high-fidelity Adams/Car model. In both cases the final results obtained showed that the correlated models were able to accurately represent the K&C behaviour of the physical vehicle, however the time investment to obtain such a correlation was significant. One of the goals of the proposed SEK joint is to allow quick transformation of K&C test data to a multibody representation of the vehicle. For studies where the suspension model is simply required in a "black-box" use-case the characteristic approach to modelling the suspension is perfectly acceptable as long as accurate dynamics are enforced by the joint. On the other hand, if specific details of the suspension system are required, such as the loading on individual suspension links or chassis mounting points, then a geometric model must be used.

2.2 Current research in path-following joints

Although not explicitly stated the suspension models discussed in Section 2.1.2 are path-following joints. Path-following constraints are commonly used in vehicle dynamics simulations for rail guided vehicles such as trains and roller coasters. Also known as spline joints (because the reference path is often defined using a spline), path-following joints have been developed for use in biomechanics research where the standard set of multibody constraints cannot be used to accurately represent the complex kinematics of human joints [12, 13].

One of the major challenges in developing a path-following joint is how to define the reference path. In early rail simulation software, the reference path was parameterized with respect to the projection of the path length onto the horizontal plane [54, 55]. This introduced the limitation that the path had to remain in the horizontal plane, or at least that vertical variations are minimal [54, 56–58]. This can be acceptable for simple train

models, but limiting if large vertical variations are required, for example if a roller coaster is under study, or if a general path-following joint is being created.

In rail simulation software packages there are two common ways to define the track. The first is to assemble the track using predefined track segments: straight, circular, and transition elements [58]. An alternative is to allow the user to input a set of control points, which can then be interpolated to generate the reference path curve. Cubic splines are a favoured interpolation approach. However this can lead to undesired oscillations in the interpolated path [59], for example if a straight track segment is followed by a curved segment. To avoid this, or at least minimise the problem, Pombo et al. [58] propose other methods for interpolation such as Akima splines [60, 61] and shape-preserving splines [62, 63]. An important property of these interpolation schemes is that the resulting curves are C^2 continuous, which is needed to ensure that the correct accelerations and reaction forces are calculated for the path-following constraint.

To define the kinematics of the path-following constraint, a moving coordinate frame needs to be associated to the curve using some form of curve framing approach. There are various options for doing this [64], but using a Frenet frame is a popular choice [56–58, 65, 66]. This fully defines the position and orientation of the body following the constraint, essentially by creating a prismatic joint on a curved path.

To improve passenger comfort and for roll over prevention when rail guided vehicles travel through a horizontal curve, the rail is banked; this is called cant [67]. Using a Frenet framing approach it is not possible to include the cant angle in the kinematic constraint formulation. Pombo et al. [57] introduced a cant angle variable to the kinematic constraint, which is defined as the angle between the desired normal vector and the osculating plane (the plane defined by the Frenet frame's normal and tangential vectors). The correct body orientation imposed by the introduction of this cant angle is handled by rotating the body about the tangential unit vector.

An issue not discussed in the previous papers is the handling of singularities. If a point of inflection is present in the curves, the standard formula used to calculate the normal vector fails. Kecskeméthy et al. [66] propose a method to overcome this using limit analysis.

Complex, path-following joints are also used to model biological joints, such as the femorotibial joint which both slides and rotates as the knee is flexed and extended [68]. Historically it has been common to model biomechanical joints using the standard set of multibody constraints to create compound joints; in [69] a framework was proposed to

create general joints using this methodology. In [12,70] the knee was modelled as a revolute joint translating along a curved path. Lee [71] developed a 1-DOF spline joint using screw theory and Lie Algebra [72]. Fan [73] expanded the work done by Lee [71] to create a 2-DOF spline joint. In this case, rather than being constrained to follow a reference path, the body is constrained to follow a reference surface.

2.3 Joint compliance

In practical applications, most systems will display some form of deformation if subjected to loads in directions that should be constrained; this is called compliance. Compliance in a passenger vehicle's suspension is primarily caused by bushings that are installed for the reduction of noise, vibration and harshness (NVH) to improve passenger comfort. A secondary cause comes from flexibility in suspension mounting points, links, bearings and joints. Compliance can have an impact on the performance of a vehicle [74] as it can cause variation in the suspension kinematics, depending on the suspension loading. This changes the orientation at which the tyre is presented to the ground, and so can be important to consider in a simulation, especially for passenger vehicles.

Blundell [75] investigated the influence of bushings on the kinematics of a multibody vehicle model. He compared three models, one with ideal kinematic constraints, another with linear bushing models and a third with non-linear bushing models. Some kinematic properties of the suspension, such as camber, were unaffected by the omission of bushings from the model; however, there was a large discrepancy in the toe angle kinematics between the suspension models with and without bushings. In all cases there was little, if any, difference between linear and non-linear bushings. Ambrósio et al. [76] demonstrated the impact of bushings on vehicle handling using an obstacle avoidance lane change maneuver. A vehicle modelled with ideal joints was compared to a vehicle modelled with bushings, and again there was a significant difference in the behaviour of the two models.

The accepted mathematical model to represent simple linear rubber bushings in a multibody system is:

$$\{F\} = [k]\{d\} + [c]\{v\} \tag{2.3}$$

where $\{F\}$ represents the forces and moments generated by the bushing, $[k]$ and $[c]$ represent the stiffness and damping of the bushing, and $\{d\}$ and $\{v\}$ represent the relative

displacement and velocities of the two connected bodies [75].

CarSim allows users to specify eight linear compliance coefficients for each wheel that represent changes in steer and camber angle due to lateral and longitudinal forces and an aligning moment, change in wheel center longitudinal position due to longitudinal forces and change in lateral position due to lateral forces [22]. The left- and right-side compliances are independent of each other; however for a steered suspension an additional non-linear steering compliance function can be defined by way of a lookup table that will add to the steer angle compliance coefficient for aligning moments. The CarSim coefficients map directly to the compliance measurements taken during K&C testing [77].

The SEK joint incorporates stiffnesses for each of the five constrained motions (two translations and three rotations). These can be mapped to a set of compliance coefficients corresponding to those reported from a K&C test. Because it is the intention that the SEK suspension model be generated from K&C test data, which is a steady state analysis (although dynamic K&C testing is now possible [78]) of the suspension, the damping compliance will not be initially applicable to the suspension joint, but a simple linear model is included for other applications.

The ability to model compliance is also important for the general application of the SEK joint as many mechanical systems incorporate rubber bushings for vibration damping, which require a compliance model to be accurately represented. Biological joints are held together with connective tissue that also displays compliance properties.

2.4 Interpolation and splines

Interpolation is the process by which "an approximating function is constructed in such a way as to agree perfectly with the usually unknown original function at the given measurement points" [79]. It is believed that the first use of interpolation was by astronomers in Ancient Babylon and Greece, to fill in gaps from missing observations [80]. This information was used for the functioning of the society, for example, by farmers who used the information to plan their planting strategies [79]. The first recorded use of interpolation is from Hipparchus of Rhodes who used linear interpolation to create tables of the "chord function" around 150BC [81]. Moving beyond linear interpolation, Liù Zhuó is thought to be the first person to use second order interpolation, around 600AD [82]. With his

work that resulted in the Newton Polynomial, Sir Isaac Newton is widely accepted to be a large driving force behind the development of interpolation methods during more recent history [83], building on the work done by Copernicus, Kepler and Gallileo [79].

When dealing with a long series that cannot be adequately represented by a simple polynomial, multiple polynomials can be created to represent the desired ranges of the data, creating a piecewise polynomial function, and what we now call a spline. Unless special attention is given to the continuity of the derivatives of these piecewise polynomials they will only be, at best, C^0-continuous. At the turn of the 19th century Karup [84] and King [85] independently discovered a third-order polynomial spline interpolant that is C^1-continuous along it's length.

Undoubtedly the pioneer of modern day spline research is Isaac Schoenberg, who in 1946 published his paper presenting the concept of basis functions and their extension to B-splines [86, 87]. A large focus of spline research during this time period was in generating splines with continuous higher-order derivatives, and with minimal oscillations, i.e. increased smoothness, without compromising the accuracy of the fit to the data points. One of the most well known piecewise polynomial fitting algorithms was developed by Hiroshi Akima [60,61]. The so-called Akima splines aim to eliminate oscillations in the spline between data points and create a spline that "appears smooth and natural" [60]. Unfortunately, the Akima splines are not C^2 continuous, which is a mandatory requirement for the construction of the SEK joint equations, and furthermore the algorithm can only be used to generate cubic splines, whereas quintic splines are used in the SEK joint. Another popular class of splines is shape preserving splines [62, 63] which have the same goal as the Akima algorithm. The disadvantage of the shape preserving splines is that they do not have compact support. Schoenberg's B-splines do not have any of these drawbacks. B-splines are easy to work with and understand because their mathematical representation is not the result of an optimization problem, as is the case with smoothing splines [88]. Due to their relative simplicity and because they are so widely researched, B-splines are used in this work.

Carl de Boor, who worked with Schoenberg at the University of Wisonsin-Madison, is credited with continuing and refining the work done by Schoenberg throughout the late

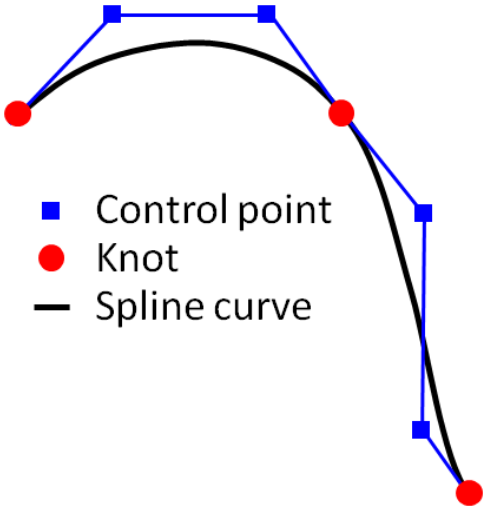


Figure 2.1: Overview of the components in a B-spline.

1900's. A B-spline [61] is described by

$$S(s) = \sum_{i=0}^{L-1} B_i^m(s, \lambda) p_i \tag{2.4}$$

where L is the number of curve pieces, B_i is the basis function, p_i are the control points, λ is the vector of knots, and m is the order of the spline. The order of the resulting polynomial is $m - 1$. B-splines are popular because they are easy to construct if the knot locations are known. The shape of the spline can be controlled using the control points (Figure 2.1). The downside of B-splines is that they can be time consuming to evaluate; however they can be easily converted to piecewise polynomial splines. In fact, Schumaker suggests that if a B-spline is to be evaluated at least twice, then it is more time-efficient to convert it to a piecewise polynomial [89].

Given a vector of knots, and the data to which the spline is to be fit, it is straight forward to determine the set of optimal control points using least squares:

$$F(p) = [x - Gp]^T [x - Gp] \tag{2.5}$$

where

$$G_{ij} = B_j^m(s_i, \lambda). \tag{2.6}$$

This approach is implemented in the MATLAB function `spap2(knots,m,s,x)`.

For any type of spline, the location and number of knots can have a large impact on the quality of fit of the resulting spline [61, 90]. Unfortunately, finding the optimal set of knots is a highly non-linear problem that contains many local minima [91]. There are two different approaches that can be taken to solve the problem of finding an optimal set of knot locations. The first uses higher order derivatives of the original data to understand the geometric properties of the curve and therefore find the optimal knot locations. Gerhard Hölze was the first to publish a method using such an approach [92]. First, given the curve $C(s)$ over the interval $[a, b]$, the function $g(t)$ is defined as

$$g(t) = \int_a^t |C^{(n+1)}(s)|^{\frac{1}{n+1}} \, \mathrm{d}s \tag{2.7}$$

where n is the order of the spline to be fit. Next, the required number of knots $(m + 1)$ can be found using:

$$m = \left\lfloor \frac{g(b)}{2} [(n+1)!\delta]^{\frac{1}{n+1}} \right\rfloor \tag{2.8}$$

where δ is the error tolerance required from the spline fit. Finally, the knot locations are chosen such that

$$g(t_i) = \frac{i g(b)}{m} \quad i = 0, \ldots, m. \tag{2.9}$$

An obvious challenge with this method when dealing with a data set for which the underlying mathematical function is unknown is the calculation of the higher-order derivative that is required to compute the function $g(t)$. For example, to fit a third order spline, the fourth derivative is required. In cases where the underlying function is unknown, estimating high-order derivatives from the numerical data can be more challenging than estimating the unknown function in the first place [93].

The second approach to finding optimal knot locations involves using computer algorithms to find the optimal knot locations using statistical measures and optimization routines. There are a number of different algorithms to solve this problem. Su and Liu developed a heuristic approach to knot finding [94]. Their algorithm looks for groupings of data points for which the interpolating curve would have locally small deflection. Knots are then placed between these groups of data points that have locally small deflection. In other words, knots are placed at points of high deflection, which intuitively makes sense. Li et al. further developed this algorithm to improve its robustness when dealing with with

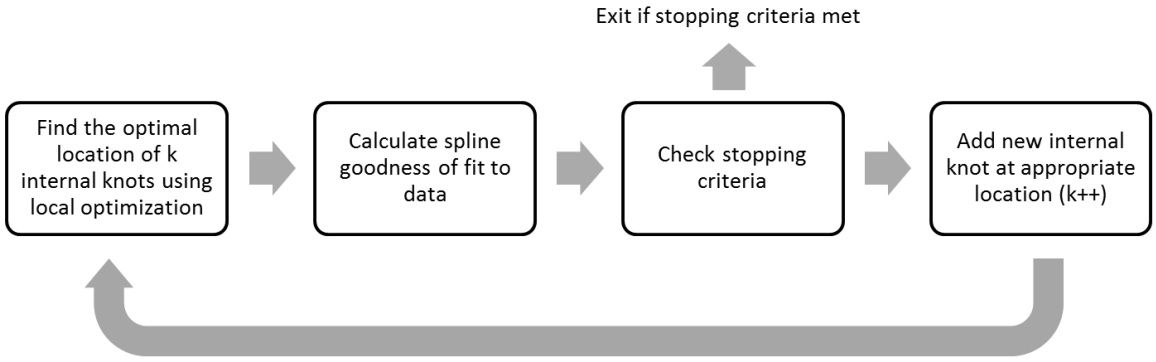


Figure 2.2: Overview of the components in a B-spline.

noisy data [95]. Their method works by applying an adaptive smoothing filter to the data when searching for the sets of points with locally small deflection. Mamic and Bennamoun proposed a probabilistic approach using Bayes Theorem [96]. They generate a model in which the random variables are the number of knots and their locations, and then estimate them using Monte Carlo simulation.

A popular and reliable method to find optimal knot locations was developed by Tao and Watson [91]. Their approach is an interactive one, in which knots are added to the curve sequentially until a stopping condition is met. This algorithm helps to find the optimal location of the minimal number of internal knots, k.

The Tao algorithm is summarized in Figure 2.2. Starting with no internal knots (i.e. $k = 0$), at each iteration of the algorithm a single knot is added. After each knot is added a local optimization algorithm is used to determine the knot locations that minimize the error between the spline curve and the path-length parameterized data.

The new knots are added with a suitable initial estimate for their location. Tao's method considers the sum of squares residual (SSR) for each segment of the spline:

$$\mathrm{SSR}_j^k = \sum_{i=1}^{n_j} [f_i - S(x_i)]^2 \tag{2.10}$$

where j is the segment number and n_j is the number of data points in the j^{th} segment. The SSR is normalized based on the total SSR for the entire curve (TSSR)

$$\text{TSSR}^k = \sum_{i=1}^{k} \text{SSR}_i^k. \tag{2.11}$$

Using this, a knot addition index (KAI) can be calculated for each segment:

$$\text{KAI}_j^k = \frac{\text{SSR}_j^k}{\text{TSSR}^k} + \frac{n_j}{np} \quad j = 1, \ldots, k+1 \tag{2.12}$$

where np is the total number of data points. The new knot is added to the center of the spline segment with the highest KAI (the largest normalized error), which provides the initial guess for the optimization algorithm. The algorithm continues until a stopping condition is met. Tao and Watson provided two stopping conditions. The first is designed to stop when the addition of more knots would have no significant impact on the goodness of the fit:

$$\frac{\text{TSSR}^0 - \text{TSSR}^{k-1}}{\text{TSSR}^0 - \text{TSSR}^k} > C \tag{2.13}$$

where C is chosen based on the desired accuracy of the spline. This allows the user to define the accuracy they expect of the spline, with the trade-off being more spline segments. The second stopping condition is a statistical measure that prevents over fitting of noisy data:

$$\left(\frac{\text{TSSR}}{np - n}\right)^k > \left(\frac{\text{TSSR}}{np - n}\right)^{k-1} \tag{2.14}$$

where n is the dimension of the spline space, the degree of continuity of the spline [61]. Other authors have proposed alternative stopping conditions, for example Lane et al. proposed using a statistical F-test as a stopping criteria to prevent over-fitting [97].

Chapter 3

Theory and software implementation

The primary goal of the single-DOF equivalent kinematic (SEK) joint is to develop a reduced suspension model that will offer a simpler modelling definition and allow faster simulation times when compared to more complex models, while maintaining the same kinematic behaviour of the equivalent high-fidelity suspension model. This will ensure that the correct forces are transmitted between the vehicle chassis and the road and the dynamic behaviour is correct. The resulting joint eliminates all closed kinematic loops and bushings that lead to high simulation times, by neglecting all suspension links and approximating the distribution of their mass properties between the unsprung and sprung masses of the vehicle. Kinematic translations and rotations are used to ensure correct wheel carrier position and orientation along a user-defined reference path. The single-DOF represents the vertical motion of the suspension, and the two constrained translations represent the lateral and longitudinal wheel center migration. The orientation is defined using three body-fixed rotations that come from three well-known vehicle suspension terms: toe, camber, and wheel spin angle. The reference path and body orientation can be defined using symbolic functions or lookup tables obtained from K&C testing or a higher-fidelity suspension model. For a steered suspension, the kinematics are also a function of the steering angle, and so a second DOF is required to model this. To handle this scenario, the SEK joint is extended to a 2-DOF version, called the double-DOF equivalent kinematics (DEK) joint. In this Chapter the development of the generalized SEK and DEK joints is presented. The joints are then validated against hand calculations, existing kinematic constraints, and simple multibody systems.

The SEK and DEK joints are developed using MapleSim and the symbolic comput-

ing language Maple. The symbolic simplification and optimization algorithms used by MapleSim to build the system equations generate dynamic simulation code that is computationally efficient. Coupling the conceptual simplicity of the SEK and DEK joints with the optimized simulation code generated by MapleSim will allow models built with the SEK and DEK joints to outperform more traditional models.

In Section 3.1 the initial formulation of the SEK joint, is presented. This formulation was not capable of representing a kinematic pair using a single ODE, and so further research was conducted to develop a reformulated joint capable of meeting this goal. In Section 3.2 the development and validation of the formulation of the ODE SEK joint is presented. This formulation allowed all of the problems with the DAE SEK joint to be overcome. An extension of the ODE SEK joint is the compliant SEK joint which is presented in Section 3.2.4. Finally, in Section 3.3 the development of the 2-DOF DEK joint is presented.

3.1 DAE SEK joint

This section discusses the first iteration of the SEK joint that was created from first principles and implemented using MapleSim. The modelling approach used to define this first version of the joint did not meet the goal of using a single ordinary differential equation (ODE) to describe the 1-DOF relative motion between the two bodies. Despite this, the work is still presented as it is useful to introduce the theory of the joint and for validation purposes. Because the formulation resulted in a set of differential-algebraic equations (DAEs) this version of the joint is referred to as the DAE SEK joint. First, a 2D particle joint is created, followed by an extension to a 3D particle. Next, a single rotation is introduced and a 2D rigid body joint is created, before moving to a general joint for a rigid body in 3D space.

3.1.1 2D particle implementation and validation

The first step in the development of the DAE SEK joint is the creation of a 2-dimensional particle joint. This implementation of the joint operates in the xz-plane. The independent variable is the z-displacement. For a given value of z, the displacement along the x-axis is constrained by the joint. This convention is chosen based on the SAE vehicle dynamics sign convention [98] in which the z-axis is vertical, considering that this joint is intended

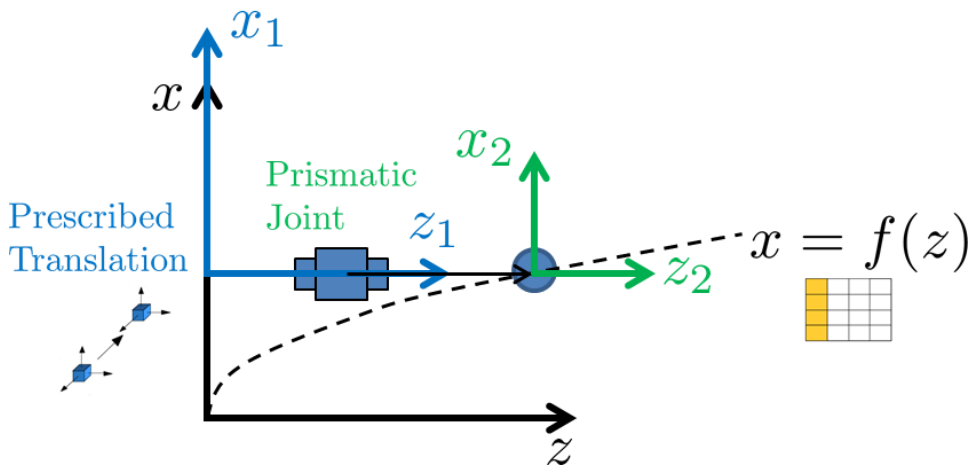


Figure 3.1: Conceptual diagram of 2D particle DAE SEK joint kinematics. x_1 and z_1 are the coordinate frame for the intermediate frame after the prescribed translation, and x_2 and z_2 are the local coordinate frame on the particle constrained by the 2D particle SEK joint.

for use in a suspension model and a suspension's DOF is generally in the vertical direction. Care must be taken to ensure that the correct reaction force between the particle and ground is applied.

The MapleSim implementation of the joint was proposed in [99], and is explained in detail here. The kinematics of the joint (Figure 3.1) are straightforward. A prismatic joint offers the unconstrained single-DOF along the z-axis. The displacement along the z-axis of this joint is used as an input to the lookup table (or a symbolic function) that calculates the displacement of the particle along the x-axis, based on the user-defined reference path, $x = f(z)$. This displacement is then applied using a prescribed translation. This is shown conceptually in Figure 3.1.

A driving constraint [100] or motion driver does not represent a DOF, it simply constrains the position of a body to a certain time-varying location, applying whatever forces are necessary. Consequently the x-component dynamics of the joint are not handled by the prescribed translation. Furthermore, by definition, a prismatic joint does not constrain motion along its DOF (the z-axis in this case), and therefore there is no z-component in the joint reaction force. In the path-following joint there will be a component of the reaction force along the z-axis. For these two reasons the MapleSim model must be expanded beyond the straightforward kinematic definition so that the reaction forces are

calculated, and the correct symbolic equations of motion for the joint consisting of both x- and z-components are generated.

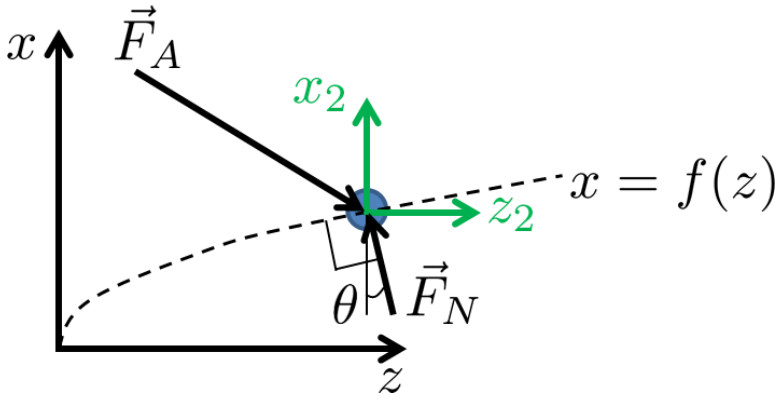


Figure 3.2: 2D particle free body diagram

The general case of a particle constrained to follow the arbitrary curve $x = f(z)$, and subjected to any external force(s), which are grouped into $\vec{F}_A$, is illustrated in Figure 3.2. The equations of motion for this system are:

$$\sum F_x : \; F_{A_x} + F_{N_x} = ma_x \tag{3.1}$$

$$\sum F_z : \; F_{A_z} + F_{N_z} = ma_z \tag{3.2}$$

To ensure the correct implementation of the joint in MapleSim the dynamics of the x- and z-components must be coupled. This logic must be added manually due to the formulation of the joint; using a prismatic joint to give the DOF along the z-axis, and a prescribed translation to enforce the correct x-displacement. To accomplish this, the following additions are made to the model; first, the x-component of the prismatic joint reaction force, F_{N_x} is sensed using a force sensor. Next, this is multiplied by the spatial derivative of the reference path, $\frac{\partial x}{\partial z}$. This gives the z-component of the reaction force, F_{N_z}, since the reaction forces in the joint must be perpendicular to the reference path. Referring to Figure 3.2, the following relationship can be developed

$$\tan\theta = \frac{\partial x}{\partial z} = \frac{F_{N_z}}{F_{N_x}}. \tag{3.3}$$

Using (3.3), F_{N_z} can be computed

$$F_{N_z} = F_{N_x}\left(\frac{\partial x}{\partial z}\right) = F_{N_x}\frac{F_{N_z}}{F_{N_x}}. \tag{3.4}$$

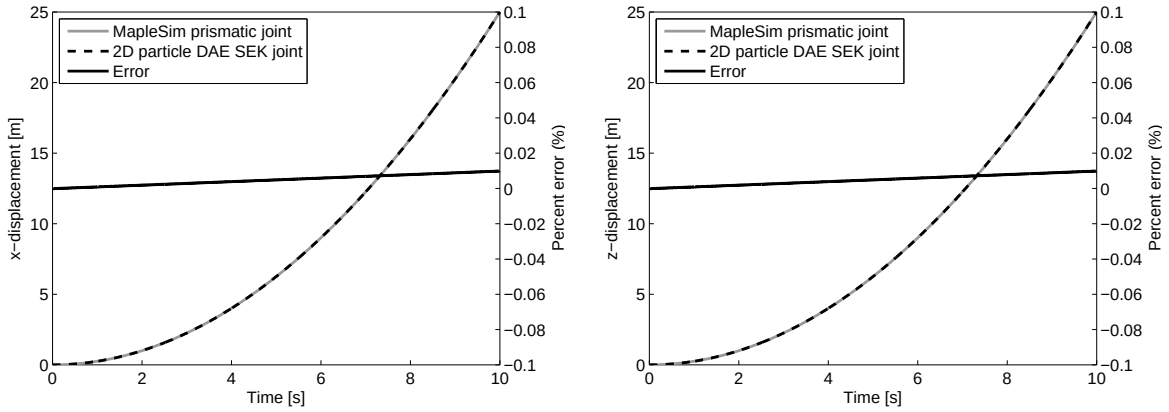
F_{N_z} is then applied along the DOF (z-axis) of the prismatic joint to ensure the correct z-component dynamics are enforced by the joint. The model is built in this way as it results in coupled x- and z-component dynamics for the joint, as desired, resulting in two ODEs in x and z being generated by MapleSim and giving the correct behaviour. Unfortunately for this implementation an algebraic equation (AE) enforcing the constraint $x = f(z)$ is required, meaning the system is still described by a small set of DAEs, rather than the desired pure ODE representation. A more detailed explanation of the MapleSim generated equations is given in Appendix A.1.

Next, the governing equations for the joint are validated against existing models. Because there is currently no path-following joint in MapleSim, the DAE SEK joint is compared with a simple prismatic joint. This is a trivial validation; however if the reaction forces were calculated incorrectly, the results of the DAE SEK joint would not match the prismatic joint. The prismatic joint is located in the xz-plane, oriented at 45° from the z-axis. This corresponds to a path equation of $x = z$ for the DAE SEK joint. Starting from rest at $z = 0$, a force of $1\hat{\mathbf{I}}$ N is applied to the particle of mass 1 kg. The MapleSim variable step RKF45 solver was used with a solver tolerance of $1e-3$. Figure 3.3 shows that under these conditions the trajectory of a particle constrained by the DAE SEK joint is the same as that of a particle constrained by a prismatic joint. The percentage error between the two models is also shown in Figure 3.3. This calculation is used extensively in this thesis when comparing two different measurements. The percentage error is calculated as

$$\text{Percent error} = \frac{\text{Measured value} - \text{Expected value}}{\text{Range of expected values}} \times 100\% \tag{3.5}$$

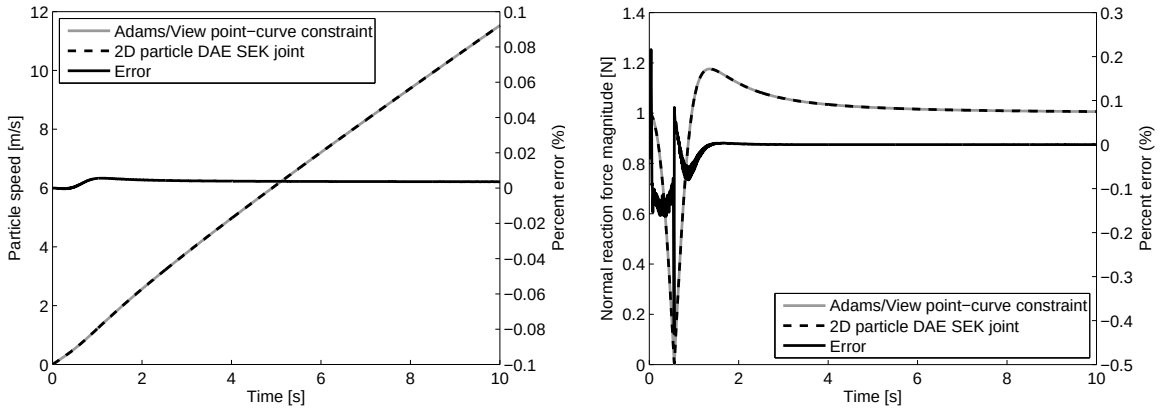
The very small error shown in Figure 3.3 is due to numerical errors that accumulate during the simulation.

A further validation can be made using the point-curve constraint in Adams/View [101]. This constraint forces a body to follow a reference path defined by a lookup table; however it does not constrain the orientation of the body, and therefore is useful to validate the particle DAE SEK joint. The reference path for both of the joints is defined as $x = z^2$. Starting from rest at $z = 0$ a force of $1\hat{\mathbf{I}} + 1\hat{\mathbf{K}}$ N is applied to a particle with a mass of 1



(a) x-displacement versus time (b) z-displacement versus time

Figure 3.3: Validation of 2D particle DAE SEK joint against MapleSim prismatic joint



(a) Speed versus time (b) Normal force magnitude versus time

Figure 3.4: Validation of 2D particle DAE SEK joint against Adams/View point-curve constraint

kg. The MapleSim variable step RKF45 and Adams/View variable step GSTIFF solvers were used for this comparison. The validation was successful and the matching velocity magnitude for the particle and reaction force in the joint are shown in Figure 3.4. The small errors are due to numerical inaccuracies in the solvers.

As a final validation of the joint it is also verified that the law of conservation of energy is obeyed. To do this, the particle is constrained to move along a parabolic reference

Table 3.1: 2D particle simulation parameters

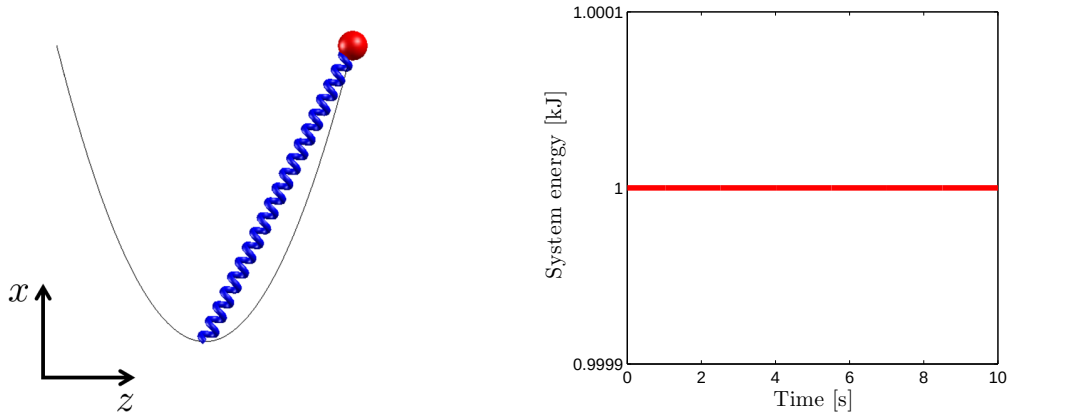
Parameter	Value
Reference path	$x = z^2$
Reference path slope	$\frac{\partial x}{\partial z} = 2z$
$m_{particle}$	1 kg
k_{spring}	100 N/m
l_0	0 m
Acceleration due to gravity, g	0 m/s^2

path, attached to the vertex of the parabola with a spring and started with an initial displacement of $z = 5m$, as shown in Figure 3.5a. Since gravity is not considered in the model, the total energy of the system is calculated as

$$E = \frac{1}{2}m_{particle}v^2 + \frac{1}{2}k_{spring}(l - l_0)^2 \tag{3.6}$$

where $m_{particle}$ is the mass of the particle, v is the speed of the particle, k_{spring} is the spring constant, l is the length of the spring, and l_0 is the unstretched length of the spring.

Figure 3.5b shows that the system energy remains constant. Table 3.1 shows the parameters used in the simulation.



(a) Energy conservation test model

(b) Results showing that energy is conserved by 2D particle DAE SEK joint

Figure 3.5: Energy conservation validation of 2D particle DAE SEK joint

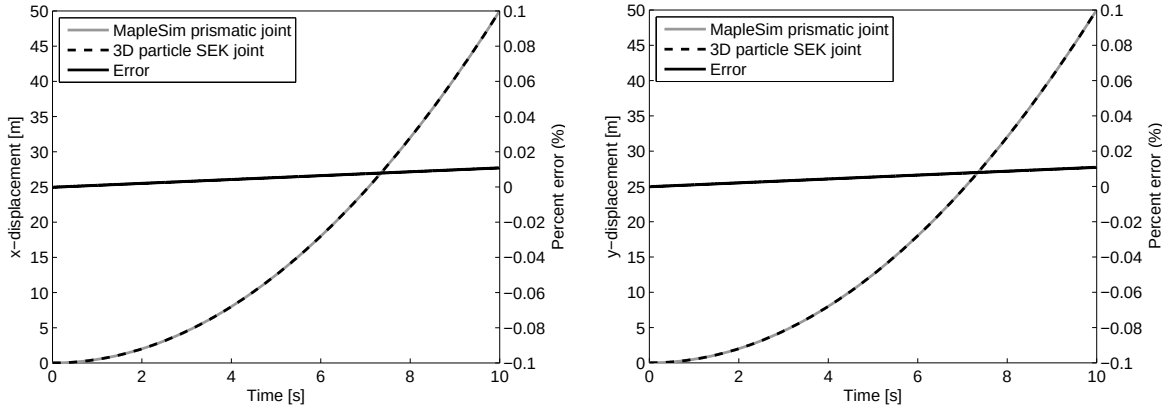
3.1.2 3D particle implementation and validation

The extension of the 2D particle DAE SEK joint to 3D is straightforward. Now both the x- and y-positions are expressed as functions of z, $x = f(z)$ and $y = g(z)$. A second reaction force calculation, using the same method as before, is now required for the additional dimension. As with the 2D particle joint, a comparison of the hand-derived equations and the MapleSim generated equations shows that the implementation appears to be correct. To validate the joint, it is successfully compared with a prismatic joint in MapleSim. The path for the SEK joint is defined by the line $x = z$, $y = z$. There is no gravitational force. The particle starts from rest at $z = 0$, and a force of $1\hat{\mathbf{I}} + 1\hat{\mathbf{J}}$ N is applied to the particle. Figure 3.6 shows that the response is the same for both the prismatic and SEK constraints.

3.1.3 2D rigid body implementation and validation

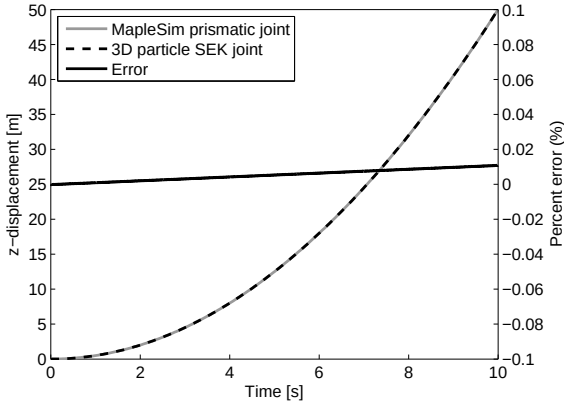
The next step in the development process is to extend the joint so it can constrain a rigid body moving along a path in the xz-plane, and rotating about its centroidal y-axis. Once a constrained body rotation is introduced, it is necessary to determine how the kinematics of the joint should be defined, specifically how the rotation is specified, and to ensure that the reaction moment is correctly calculated and projected onto the motion space. One approach is to determine the body orientation about the y-axis using the path definition, so that the body axes are aligned with the normal and tangential vectors of the path [102]. This implementation of the joint can be imagined as a prismatic joint that has been bent to follow a user-defined path over a flat surface. The second option is to allow the rotation about the y-axis to be determined as a more general function of the displacement along the z-axis, $\theta = h(z)$. This is the implementation that was selected. The kinematics are a straightforward addition to the particle joint; a prescribed rotation is used to orient the body after the translations are applied as shown in Figure 3.7.

The general case of a rigid body constrained to follow the curve $x = f(z)$ with the body orientation $\theta = h(z)$, subjected to external force(s), grouped into $\vec{F}_A$, and moment(s), grouped into $\vec{M}_A$, and with reaction force $\vec{F}_N$ and moment $\vec{M}_R$ is illustrated in Figure 3.8. The translational dynamics are still governed by (3.1) and (3.2). As with the translational dynamics presented in Section 3.1.1, the first step in understanding the rotational dynamics is to derive the equation of motion. The total moment about the y-axis through the center



(a) x-displacement versus time

(b) y-displacement versus time



(c) z-displacement versus time

Figure 3.6: Validation of 3D particle SEK joint against prismatic joint

of mass can be calculated as

$$\sum M_y : M_R + M_A = I_{yy}\alpha_y \tag{3.7}$$

where M_R is the reaction moment in the joint about the y-axis, M_A is the known, applied moment about the y-axis, I_{yy} is the moment of inertia about the centroidal y-axis and α_y is the angular acceleration of the body about the y-axis.

If lookup tables are used to define the path, as is the case in this example, the first spatial derivative of the body orientation function must be defined in an additional lookup table. Ensuring the correct dynamic response of the joint to applied and reaction moments

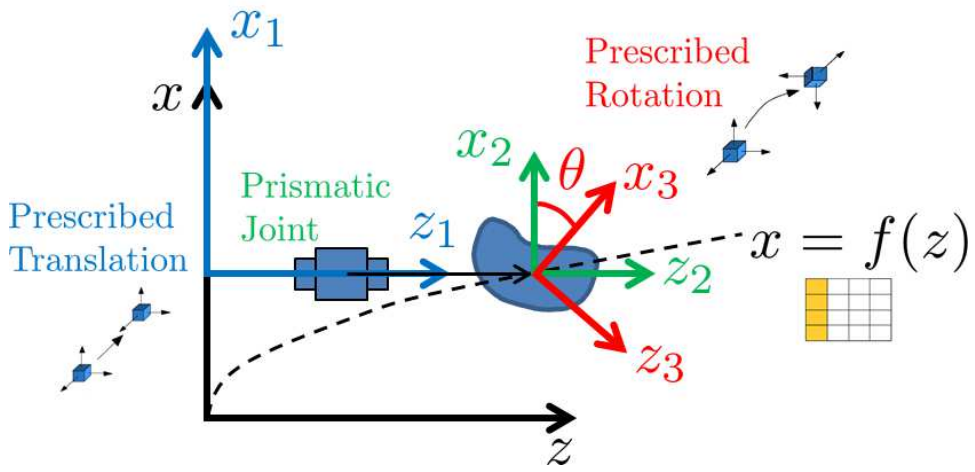


Figure 3.7: Conceptual diagram of 2D rigid body DAE SEK joint kinematics. x_1 and z_1 are the coordinate frame for the intermediate frame after the prescribed translation, and x_2 and z_2 are the intermediate local coordinate frame on the rigid constrained by the SEK joint before the rotation is applied, x_3 and z_3 are the local coordinate frame on the rigid body after the rotation θ is applied.

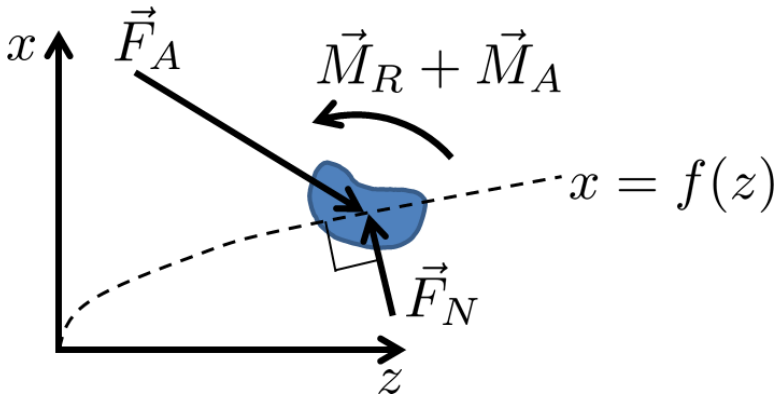


Figure 3.8: 2D rigid body free body diagram

is accomplished using a similar coupling approach to that used for the reaction forces in the particle joints discussed previously. First, the reaction moment between the prismatic joint and the rigid body about the y-axis is measured using a force and moment sensor. Once this is done, the y-component of the reaction moment, M_R, is multiplied with the spatial derivative of the body orientation, $\frac{\partial \theta}{\partial z}$, to determine the force that should be applied to the prismatic joint along the z-axis (F_{θ_z}) to couple the rotational dynamics with the translational dynamics. Using an approach similar to that presented in (3.4), the following

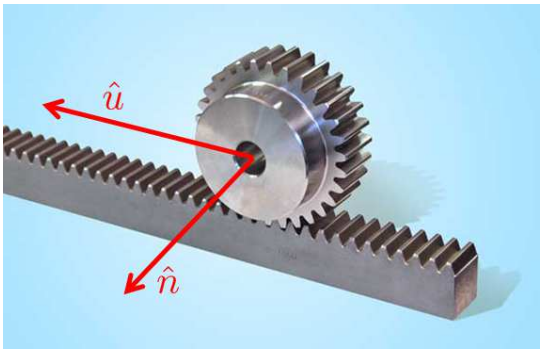


Figure 3.9: Rack and pinion system

relationship is developed

$$\frac{\partial \theta}{\partial z} = \frac{F_{\theta_z}}{M_R}. \tag{3.8}$$

This can be used to compute the correct value for F_{θ_z}, the force that should be applied along the motion space of the prismatic joint (the z-axis) to couple the translational and rotational dynamics

$$F_{\theta_z} = M_R \left(\frac{\partial \theta}{\partial z} \right) = M_R \left(\frac{F_{\theta_z}}{M_R} \right). \tag{3.9}$$

A unit analysis shows the derivation is consistent

$$[\mathrm{N}] = [\mathrm{Nm}] \left[\frac{1}{\mathrm{m}} \right] = [\mathrm{Nm}] \left[\frac{N}{\mathrm{Nm}} \right]. \tag{3.10}$$

The need for this coupling can be conceptually visualized using a rack and pinion system, shown in Figure 3.9. This type of system consists of a linear gear (the rack) along which runs a circular gear (the pinion). Considering the rack as rigidly attached to the inertial frame of reference, the pinion travels along the rack, both rotating and translating. Assuming the gear ratio is constant this type of system could be represented by the 2D rigid body DAE SEK joint in which the spatial derivative of the body orientation function ($\frac{\partial \theta}{\partial z}$) is constant. This derivative is the gear ratio. In this system, the pinion can be made to translate along the rack by applying either a force with a non-zero component along the rack (z), or by applying a moment with a non-zero component about the axis of rotation of the pinion (y). Without the dynamic coupling in the standard MapleSim model, the prismatic joint will allow motion to occur for the first case. In the second case the applied moment would be absorbed by the prismatic joint and no motion would occur. It is for

this reason that the described coupling is required. The concept of torque projection is revisited in greater detail in Section 3.2 when the final formulation of the SEK joint is presented.

For further validation the angular acceleration of the rigid body about its centroidal y-axis, $\alpha_y = \ddot{\theta}$, for a parabolic reference path, and with the body orientation constrained to remain aligned with the path can be derived. To solve for the angular velocity, first the equation describing the angular velocity is determined and then differentiated. Letting $\square'$ represent $\frac{\partial \square(z)}{\partial z}$ and using the reference path $\vec{r}(z) = \langle f(z), g(z), z \rangle$, the unit tangent vector can be calculated [102]:

$$\hat{\mathbf{u}}(z) = \frac{\vec{r'}(z)}{\|\vec{r'}(z)\|} \tag{3.11}$$

where

$$\vec{r'}(z) = \frac{\partial f(z)}{\partial z}\hat{\mathbf{i}} + \frac{\partial g(z)}{\partial z}\hat{\mathbf{j}} + 1\hat{\mathbf{k}}. \tag{3.12}$$

$$\|\vec{r'}(z)\| = \sqrt{f'^2 + g'^2 + 1} \tag{3.13}$$

therefore

$$\hat{\mathbf{u}}(z) = \frac{1}{\sqrt{f'^2 + g'^2 + 1}}(f'\hat{\mathbf{i}} + g'\hat{\mathbf{j}} + 1\hat{\mathbf{k}}) \tag{3.14}$$

and

$$\dot{\hat{\mathbf{u}}}(z) = \frac{1}{(f'^2 + g'^2 + 1)^{\frac{1}{2}}}(\dot{f'}\hat{\mathbf{i}} + \dot{g'}\hat{\mathbf{j}}) - \frac{f'\dot{f'} + g'\dot{g'}}{(f'^2 + g'^2 + 1)^{\frac{3}{2}}}(f'\hat{\mathbf{i}} + g'\hat{\mathbf{j}} + 1\hat{\mathbf{k}}). \tag{3.15}$$

If the reference path is defined as $x = f(z) = z^2$, $y = g(z) = 0$, then

$$\hat{\mathbf{u}}(z) = \frac{1}{\sqrt{4z^2 + 1}}(2z\hat{\mathbf{i}} + 1\hat{\mathbf{k}}) \tag{3.16}$$

and

$$\dot{\hat{\mathbf{u}}}(z) = \left(\frac{2\dot{z}}{(4z^2 + 1)^{\frac{1}{2}}} - \frac{8z^2\dot{z}}{(4z^2 + 1)^{\frac{3}{2}}}\right)\hat{\mathbf{i}} - \frac{4z\dot{z}}{(4z^2 + 1)^{\frac{3}{2}}}\hat{\mathbf{k}}. \tag{3.17}$$

Given that [102]

$$\dot{\hat{\mathbf{u}}} = \vec{\omega} \times \hat{\mathbf{u}} \tag{3.18}$$

it can be shown that

$$\vec{\omega} = \hat{\mathbf{u}} \times \dot{\hat{\mathbf{u}}} \tag{3.19}$$

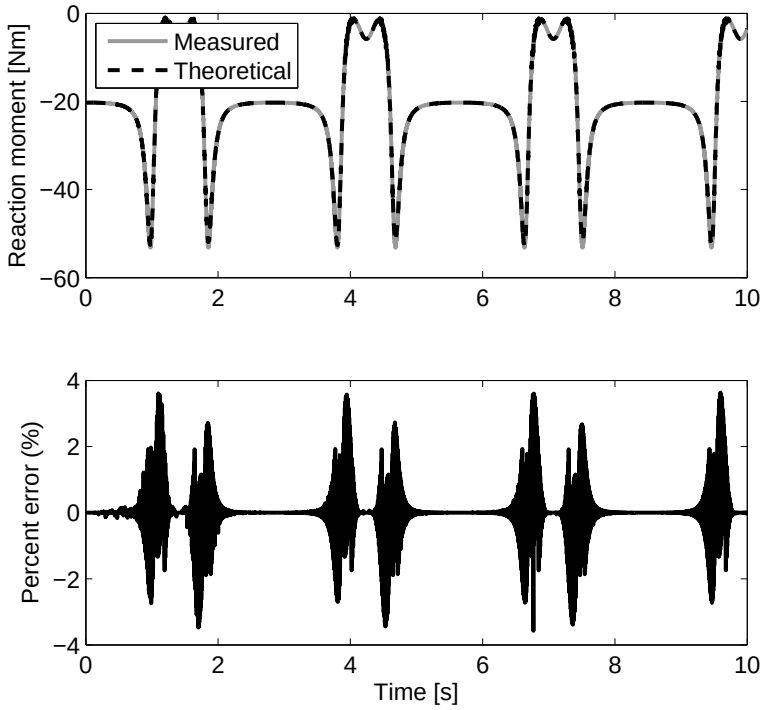


Figure 3.10: Validation of reaction moments in 2D rigid body DAE SEK joint for a parabolic path

as follows

$$\begin{aligned} \dot{\hat{\mathbf{u}}} &= \vec{\omega} \times \hat{\mathbf{u}} \\ \hat{\mathbf{u}} \times \dot{\hat{\mathbf{u}}} &= \hat{\mathbf{u}} \times \vec{\omega} \times \hat{\mathbf{u}} \\ &= \hat{\mathbf{u}} \cdot \hat{\mathbf{u}}\vec{\omega} - \hat{\mathbf{u}} \cdot \vec{\omega}\hat{\mathbf{u}} \\ &= \vec{\omega} \end{aligned} \tag{3.20}$$

since $\hat{\mathbf{u}} \cdot \hat{\mathbf{u}} = 1$ and $\hat{\mathbf{u}} \cdot \vec{\omega} = 0$.

Substituting in the equations for $\hat{\mathbf{u}}$ and $\dot{\hat{\mathbf{u}}}$ gives

$$\vec{\omega} = \frac{2\dot{z}}{4z^2 + 1}\hat{\mathbf{j}} \tag{3.21}$$

which gives

$$\dot{\vec{\omega}} = \vec{\alpha} = \left(\frac{2\ddot{z}}{4z^2 + 1} - \frac{16z\dot{z}^2}{(4z^2 + 1)^2} \right) \hat{\mathbf{j}} \tag{3.22}$$

so therefore

$$\alpha_y = \left(\frac{2\ddot{z}}{4z^2+1} - \frac{16z\dot{z}^2}{(4z^2+1)^2} \right). \tag{3.23}$$

α_y is constrained by the kinematics of the joint, meaning that for a known applied moment it is possible to calculate the correct reaction moment and compare it with the 2D rigid body DAE SEK joint. Simulating a rigid body of mass 1 kg, and I_{yy} of 1 kg m^2, subject to gravity in the negative x direction, and an applied moment about the y-axis of -20 Nm, the reaction moment is measured in MapleSim. The z position, speed and acceleration are also measured, and used in (3.23) to compute the reaction moment using (3.7). The comparison between the measured and theoretical reaction moment is shown in Figure 3.10. The two numbers match well, although there is some noise in the theoretical value as the measured acceleration is used in the calculation. The reference path has been defined using the MapleSim look up tables that use only C^1-continuous curve fits.

Table 3.2: 2D rigid body simulation parameters

Parameter	Value
Reference path	$x = z^2$
Reference path slope	$\frac{\partial x}{\partial z} = 2z$
Body orientation	$\theta = \arctan(2z)$
Body orientation derivative	$\frac{\partial \theta}{\partial z} = \frac{2}{1+4z^2}$
m_{body}	1 kg
I_{yy}	1 kg m^2
k_{spring}	100 N/m
l_0	0 m
g	0 m/s^2

As in Section 3.1.1 a test is done to ensure that energy is conserved in a system using the 2D rigid body DAE SEK joint. Because a rigid body is now considered, the rotational kinetic energy must be considered and so the energy of a system implementing the 2D rigid body joint is measured according to

$$E = \frac{1}{2} m_{body} v^2 + \frac{1}{2} I_{yy} \omega^2 + \frac{1}{2} k_{spring} (l - l_0)^2. \tag{3.24}$$

As in Section 3.1.1, the spring is attached between the body CG and the origin, and the reference path is parabolic (Figure 3.5a). The body orientation is constrained to follow the

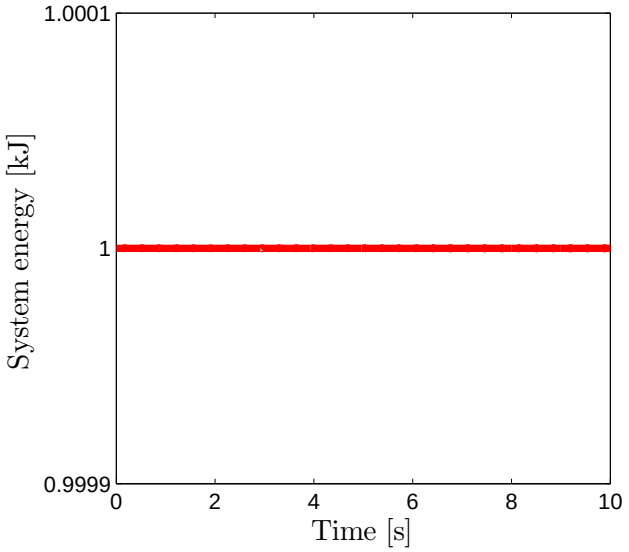


Figure 3.11: Energy conservation validation of 2D rigid body DAE SEK joint

shape of the curve. Again, energy is conserved as shown in Figure 3.11. The simulation parameters are shown in Table 3.2.

Because there is no equivalent joint in MapleSim or Adams/View with which to compare the 2D rigid body joint, it is compared with an ideal planar four-bar linkage in Adams/View, as shown in Figure 3.12. This is done by representing a single coupler bar, ①, as a rigid body in MapleSim constrained to move along a reference path by the 2D rigid body DAE SEK joint. The kinematic information from the Adams/View four-bar is used to define the reference-path and body orientation functions in the SEK joint. Along with the reference-path and body orientation, the spatial derivatives of the two curves are calculated numerically in Adams/View and exported to a lookup table in MapleSim. The four-bar is considered ideal because all links except the coupler are massless, meaning that the models should behave identically. The coupler bar is modeled with uniform mass distribution, meaning that the CG is at its geometric center.

A limitation introduced by defining the reference path and body orientation as functions of the displacement along the z-axis is that the functions must be true functions; that is, each input (z-displacement) is mapped to exactly one output [103]. This means, for example, that a circular path cannot be defined. Removing this limitation is part of the reformulation work presented in Section 3.2. Because of this limitation, the simulations of

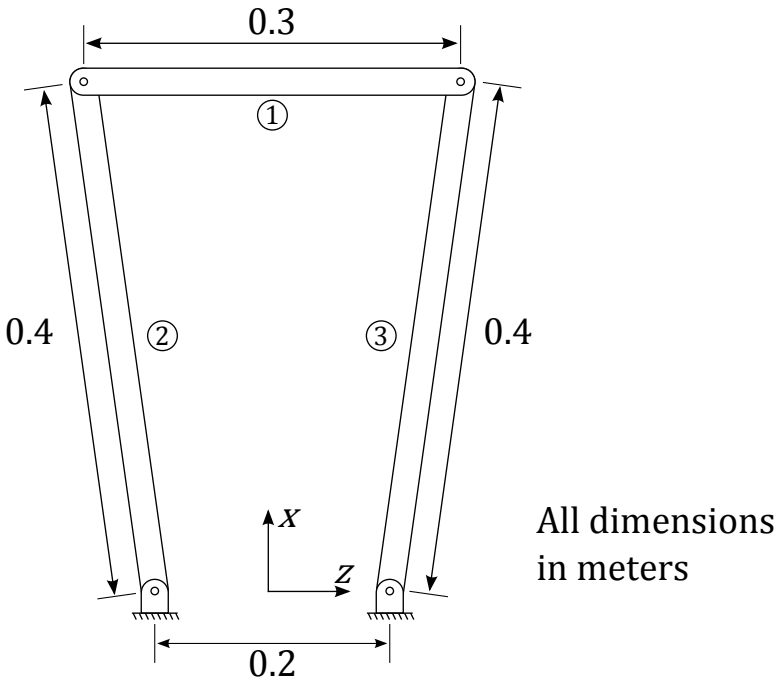


Figure 3.12: Planar four-bar dimensions

the four-bar linkage can only include a limited range of motion of the system.

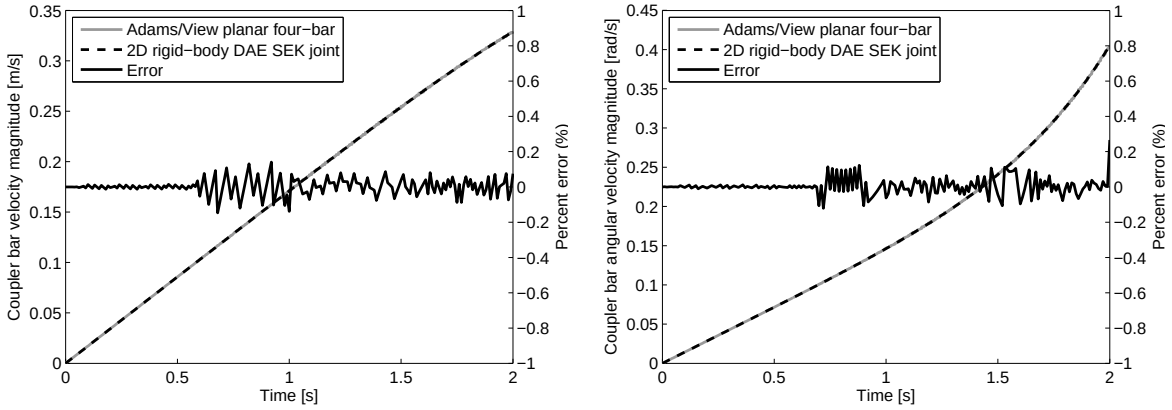
The base of the Adams/View four-bar linkage is locked to the ground using a fixed joint. Using this fixed joint makes it possible to determine the reaction forces at the ground and compare them with the DAE SEK joint reactions for validation purposes. The parameters used in the simulation can be seen in Table 3.3. Starting at rest from $x = 0.4$ m and $z = 0$ m, a force and moment are applied at the CG of bar ①, which is in the geometric center of the bar.

The speed and angular speed of the bar and the magnitude of the reaction forces and moments at the ground in the Adams/View four-bar and the representative MapleSim DAE SEK joint are shown in Figure 3.13. The results match well, providing a validation of the 2D rigid body DAE SEK joint. There is some noise in the reaction force magnitude (Figure 3.13c) and the reaction moment magnitude (Figure 3.13d). This is caused by the MapleSim lookup tables used to define the reference path, which only use a C^1 continuous spline curve fit. This problem can be overcome by using a symbolic definition of the reference path, rather than a lookup table. This is addressed with the reformulation of the SEK joint that is presented in Section 3.2.

Table 3.3: Planar four-bar simulation parameters

Parameter	Value
m_1	10 kg
m_2	0 kg
m_3	0 kg
I_{yy1}	1 kg m^2
I_{yy2}	0 kg m^2
I_{yy3}	0 kg m^2
$F_{Applied}$	$1\hat{\mathbf{i}}$ N
$M_{Applied}$	$1\hat{\mathbf{j}}$ Nm
g	0 m/s^2

This shows the first example of the SEK joint being used for model reduction. The planar four-bar when represented in Adams/View consists of three rigid bodies, and four revolute joint constraints. The SEK joint representation uses only one rigid body and one SEK joint constraint, with kinematic information from the Adams/View model, to represent the same system. If only the kinematics of the coupler bar are of interest then neglecting the other linkages in the system will not cause any important information to be lost. Since this model is only planar, the next stage of the development is to introduce additional complexity to the joint to handle 3D rotations.



(a) Speed versus time

(b) Angular speed versus time

(c) Reaction force magnitude versus time

(d) Reaction moment magnitude versus time

Figure 3.13: Validation of 2D rigid body DAE SEK joint against Adams/View planar four-bar

3.1.4 3D rigid body implementation and validation

The final step in the development process is to implement the joint for a rigid body moving along a path in 3D space. As with the 2D case, there are multiple options for defining the orientation of the body. Two of the options correspond to the definitions described for the 2D case: 1) the body orientation is aligned with the path; and 2) the body orientation is defined by the user as a function of the displacement along the z-axis. This second implementation of the joint is what is of interest for application in the suspension model. An intermediary step in the development process was a third kinematic definition in which a single rotation about the tangential unit vector is defined as a function of the path length, resulting in a screw joint.

The first implementation followed the work done in [66] using Frenet frames to determine the body orientation based on the reference path. First, the tangential unit vector, $\hat{\mathbf{u}}$, is calculated using (3.11). Next, the normal unit vector, $\hat{\mathbf{n}}$, is calculated [102] using

$$\hat{\mathbf{n}} = \frac{\rho}{s'^4}\{\vec{r''}s'^2 - \vec{r'}(\vec{r'} \cdot \vec{r''})\} \tag{3.25}$$

where s is the path length and:

$$s' = \frac{\partial s}{\partial z} = \|\vec{r'}(z)\| \tag{3.26}$$

and the radius of curvature, ρ, is:

$$\rho = \frac{1}{s'^4}\sqrt{(\vec{r''} \cdot \vec{r''})s'^2 - (\vec{r'} \cdot \vec{r''})^2}. \tag{3.27}$$

Finally, the binormal unit vector, $\hat{\mathbf{b}}$, can be calculated as

$$\hat{\mathbf{b}} = \hat{\mathbf{u}} \times \hat{\mathbf{n}}. \tag{3.28}$$

Using these unit vectors, a rotation matrix can be assembled to correctly orient the body as it travels through 3D space aligned with the reference path

$$\Big[R\Big] = \Big[\{\hat{\mathbf{u}}\} \quad \{\hat{\mathbf{n}}\} \quad \{\hat{\mathbf{b}}\}\Big] \tag{3.29}$$

where each column vector is composed of the components of the tangential, normal and binormal unit vectors expressed in the global frame. These equations were coded into a

MapleSim custom component, meaning that this implementation of the joint only accepts a reference path defined by symbolic functions, not lookup tables. Because this wasn't the desired final functionality of the SEK joint, and because there were no other joints available with which to easily compare it, this joint was superceded by the screw joint implementation of the DAE SEK joint.

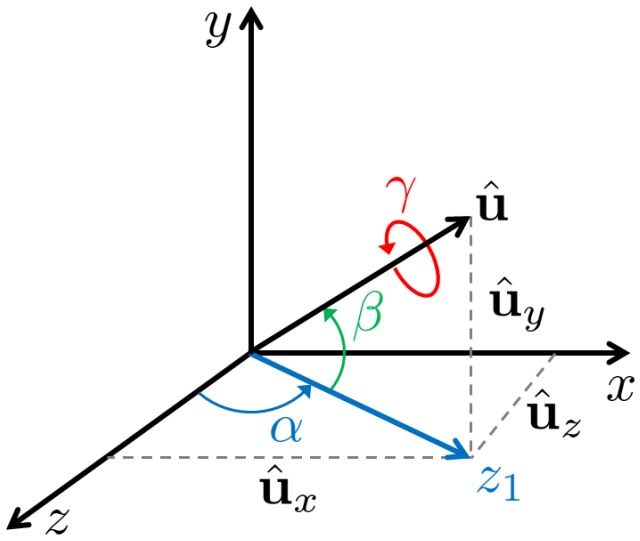


Figure 3.14: Kinematics of 3D rigid body DAE SEK screw joint. The x_1, y_1, z_1 frame is obtained by rotating the initial frame by the angle α about the y-axis.

The screw joint realization of the DAE SEK joint allows the screw angle, γ, to be specified as an arbitrary function of the path length [104], s, i.e. $\gamma = \gamma(s)$ where

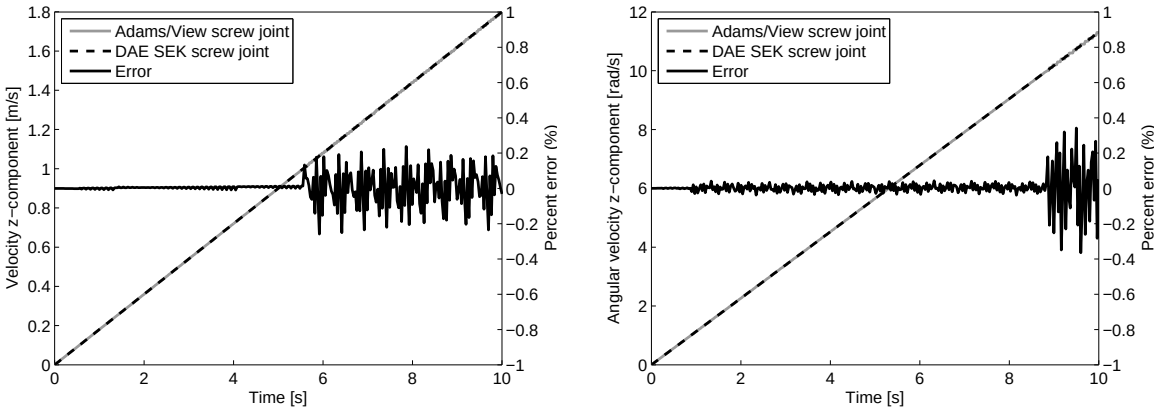
$$s = \int_0^z \sqrt{f'^2 + g'^2 + 1}\,\mathrm{d}z \tag{3.30}$$

where $\square'$ represents $\frac{\partial \square(z)}{\partial z}$. Euler angles are being used to represent the 3D orientation of the constrained body, and so three angles are needed to fully define the orientation of the body. The first is γ, and the other two are α and β which are shown in Figure 3.14, and can be determined by

$$\alpha = \arctan\left(\frac{u_x}{u_z}\right) \tag{3.31}$$

$$\beta = \arcsin(u_y). \tag{3.32}$$

As before, these angles are calculated in a MapleSim custom component and used as inputs



(a) Speed versus time

(b) Angular speed versus time

Figure 3.15: Validation of DAE SEK screw joint against Adams/View screw joint

to a prescribed rotation, with a 2-1-3 Euler angle rotation of α about the y-axis, $-\beta$ about the x_1-axis and γ about the $\hat{\mathbf{u}}$ unit vector, as shown in Figure 3.14. This implementation of the DAE SEK joint can be compared to the Adams/View screw joint [101], as long as the reference path is a straight line and the screw angle varies constantly along the path, i.e. it has a constant pitch. The pitch is defined as the amount of translation along the joint axis for each full rotation about the joint axis. Since this implementation of the joint travels on a straight path only, the reaction moment about the tangential unit vector ($\hat{\mathbf{u}}$) must be coupled to the translational dynamics using the exact same approach used in the 2D rigid body implementation to ensure the joint behaves correctly. The reaction moment is sensed using a torque sensor, and (3.9) is used to compute the appropriate coupling. In this case the derivative is $\frac{\partial \gamma}{\partial s}$ which represents the pitch of the screw joint. Starting from rest at $z = 0$, a force an moment are applied to the rigid body, to move it along the screw joint. The DAE SEK screw joint was compared with the Adams/View screw joint, and the results in Figure 3.15 show that the response is the same in MapleSim and Adams/View. The parameters used in the simulation are shown in Table 3.4.

The previous models have been validated, and resulted in an understanding of how a general DAE SEK joint should be created. The final implementation of the 3D rigid body DAE SEK joint allows the position of a rigid body to be fully specified using a reference path defined by $x = f(z)$ and $y = g(z)$, and the orientation of the body to be defined as a function of the displacement along the global z-axis using three body-fixed Euler angles.

Table 3.4: SEK screw joint validation simulation parameters

Parameter	Value
Reference path	$x = 0$, $y = 0$
Screw pitch	1 m/revolution
m_{body}	1 kg
I_{zz}	1 kg m^2
$F_{applied}$	$1\hat{\mathbf{K}}$ N
$M_{applied}$	$1\hat{\mathbf{K}}$ Nm

The translational kinematics and reaction force calculations are identical to those found in Section 3.1.2. The rotational kinematics and reaction force calculations follow the same theory as the 2D rigid body DAE SEK joint discussed in Section 3.1.3, except (3.9) must be extended to facilitate the additional rotations required for a 3D representation. Letting θ, ϕ and ψ represent the three user input functions that define the rigid body orientation using Euler angles, the coupling force that must be applied along the prismatic joint's motion space (i.e. the z-axis) can be computed as

$$F_z = M_{R_\theta}\left(\frac{\partial\theta}{\partial z}\right) + M_{R_\phi}\left(\frac{\partial\phi}{\partial z}\right) + M_{R_\psi}\left(\frac{\partial\psi}{\partial z}\right) \tag{3.33}$$

where $M_{R_\square}$ is the reaction moment about the axis of rotation for the related Euler angle. The Euler angle functions are user inputs, and are functions of z. Note that there is not a single moment sensor used, but one for each rotation, as it is important to ensure that the moments are calculated in the correct coordinate frame orientation. In this implementation, 3-1-3 body fixed Euler angles are used to define the body orientation; however the joint will work with any ordering. In the lookup table implementation of the joint, ten lookup tables are required: two for the reference path (x and y position), two for the reference path derivatives (x' and y'), three for the Euler angles, and three more for their derivatives with respect to the z-displacement.

To validate the general DAE SEK joint, it is compared to a spatial four-bar mechanism developed in Adams/View, as shown in Figure 3.16. As before, the four-bar mechanism is ideal, meaning that only the coupler bar, ①, has mass. The results in Figure 3.17 of the spatial four-bar falling under the force of gravity from $z = 0.2$ m show that the DAE SEK joint matches the response of the Adams/View four-bar well. As with the planar four-bar

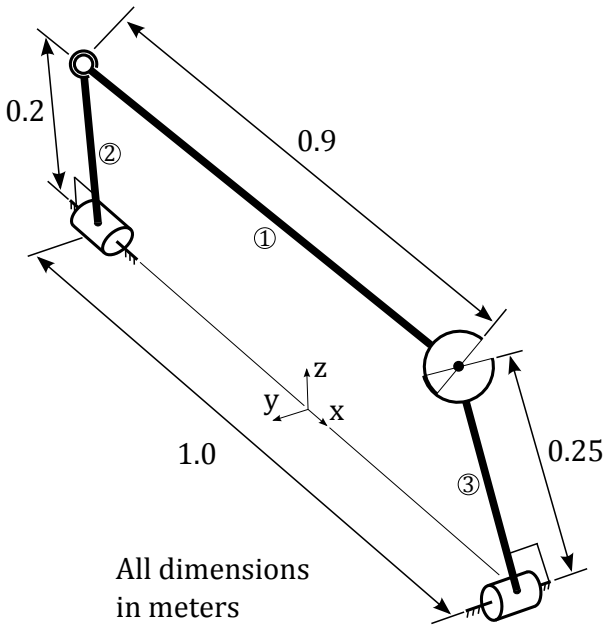


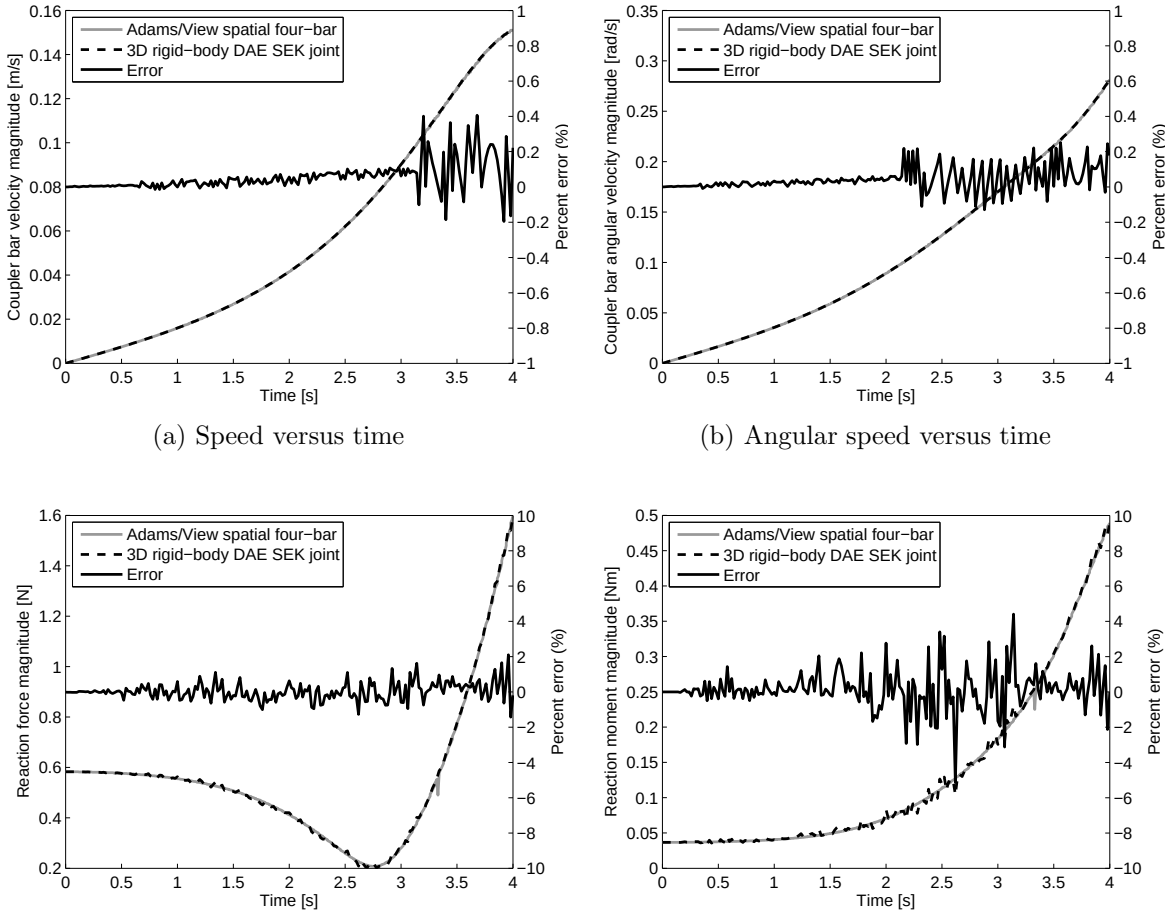
Figure 3.16: Spatial four-bar dimensions

validation in Section 3.1.3 there is some noise in the reaction force and moment magnitudes, caused by the lookup tables. The simulation parameters are shown in Table 3.5.

Table 3.5: Spatial four-bar simulation parameters

Parameter	Value
m_1	6 kg
m_2	0 kg
m_3	0 kg
$[I]_1$	[1] kg m^2
$[I]_2$	[0] kg m^2
$[I]_3$	[0] kg m^2
g	$-0.1\hat{\mathbf{K}}$ m/s^2

Although this initial version of the DAE SEK joint works well enough to prove the concept, there is a significant drawback in the implementation. By failing to eliminate the algebraic constraints in the resulting system of equations, a major goal of this new joint has not been achieved. In the next section, a revised joint is created to address this.



(a) Speed versus time

(b) Angular speed versus time

(c) Reaction force magnitude versus time

(d) Reaction moment magnitude versus time

Figure 3.17: Validation of 3D rigid body DAE SEK joint against Adams/View spatial four-bar

3.2 ODE SEK joint

In this section a revised formulation of the SEK joint is presented. Since this new formulation results in the kinematic pair being described by a single ODE (as desired) it is referred to as the ODE SEK joint. First, in Section 3.2.1 the theory of the ODE SEK joint is discussed, and some simple examples are shown along with comparisons with the DAE SEK joint formulation to highlight the differences. Next, in Section 3.2.2 the path generation algorithm is shown. This algorithm is responsible to accept as an input user-defined numerical data describing the reference path, and to output spline functions that can be used in the ODE SEK joint. In Section 3.2.3 the ODE SEK joint formulation is validated using the previously validated DAE SEK joint and conventional models in Adams/View. All of the joints developed until this point have been rigid, ideal joints. In Section 3.2.4 an extension of the ODE SEK joint is presented that introduces the concept of compliance to the SEK joint concept.

3.2.1 Joint theory

To eliminate the algebraic constraints in the mathematical description of the kinematic pair, a different formulation of the joint is required. This new formulation is described here with reference to Figure 3.18. The goal of the SEK joint is to constrain one body to move along a reference path, relative to another body. An important aspect of the SEK joint is the definition of the reference path. In this reformulation the distance along the reference path, the path-length (s), is chosen as the independent coordinate for the joint. Doing so allows the components of the position vector ($\vec{r}_{J/I}$ – i.e. the reference path) and the three Euler angles (θ, ϕ, ψ) used to orient the body to be expressed as functions of s:

$$\vec{r}_{J/I}(s) = r_x(s)\hat{\mathbf{i}} + r_y(s)\hat{\mathbf{j}} + r_z(s)\hat{\mathbf{k}} \tag{3.34}$$

$$\{\theta(s), \phi(s), \psi(s)\} = \{S_\theta(s), S_\phi(s), S_\psi(s)\}. \tag{3.35}$$

This offers the additional advantage over the DAE SEK joint that now any path can be represented by the ODE SEK joint, including those that cannot be represented by true functions [103] where a single input is mapped to non-unique outputs, such as a circle.

MapleSim uses multibody graph theory to build the equations of motion for a system [27]. The formulation of the ODE SEK joint must be compatible with the approach

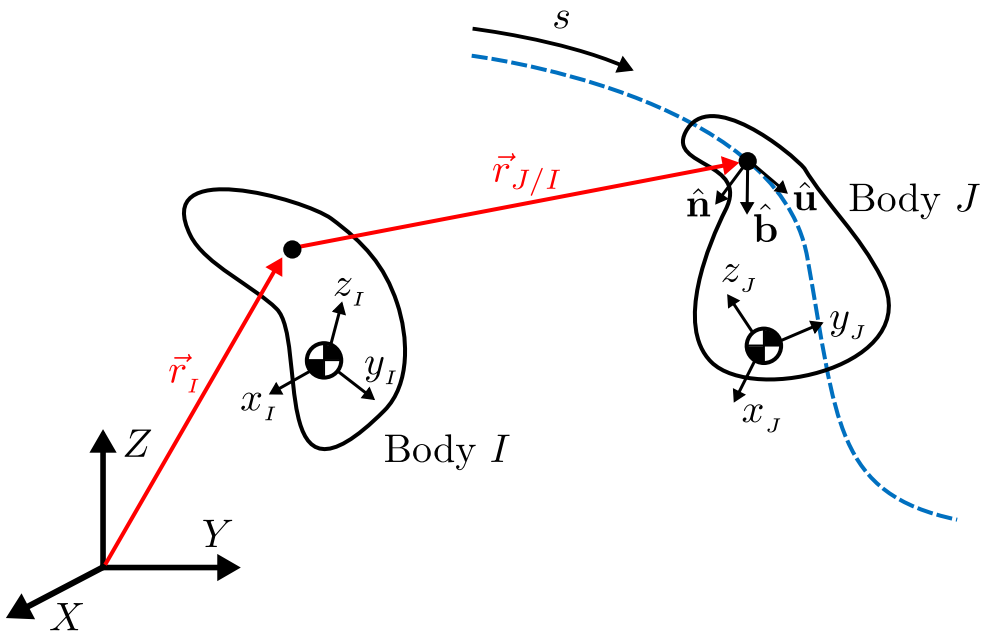


Figure 3.18: General kinematics of ODE SEK joint

used in MapleSim. In multibody graph theory each joint is represented as an edge in the graph, and has associated constitutive equations. The equations define the kinematics and dynamics enforced by the joint, such as the relative positions and velocities of the two constrained bodies. These are derived from simple kinematic relationships [102]. Two other components of the terminal equations are the motion space ($\mathcal{M}$) and reaction space ($\mathcal{F}$) of the joint. The motion space defines the motions allowed by the joint, and the reaction space defines the motions that are constrained by the joint, resulting in reaction loads. In multibody dynamics there are two domains: translational (T) and rotational (R). As an example, given a simple prismatic joint with its single DOF along the x-axis, the translational motion space is:

$$\mathcal{M}_T = \hat{\mathbf{i}}, \tag{3.36}$$

and the translational reaction space is:

$$\mathcal{F}_T = \langle \hat{\mathbf{j}}, \hat{\mathbf{k}} \rangle. \tag{3.37}$$

Because the prismatic joint offers no rotational DOF, the rotational motion space is the

empty set:

$$\mathcal{M}_R = \emptyset, \tag{3.38}$$

and the rotational reaction space is:

$$\mathcal{F}_R = \langle \hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}} \rangle. \tag{3.39}$$

To develop the ODE SEK joint, the motion and reaction spaces need to be defined. Since the reference path is arbitrary, the motion and reaction spaces may not be constant along the path, therefore the functions describing them need to be functions of s. $\mathcal{M}_T$ is the tangential unit vector ($\hat{\mathbf{u}}$) and $\mathcal{F}_T$ is defined by the plane created by the normal ($\hat{\mathbf{n}}$) and binormal ($\hat{\mathbf{b}}$) unit vectors in a Frenet frame. Defining the position vector as a function of the path length allows use of the Frenet-Serret formulas [102] to easily determine these unit vectors. Given a position vector, $\vec{r}_{J/I}(s)$, the tangential vector, $\hat{\mathbf{u}}(s)$, can be calculated and used to describe the motion space of the joint

$$\hat{\mathbf{u}}(s) = \frac{d\vec{r}_{J/I}(s)}{ds} = \frac{dr_x(s)}{ds}\hat{\mathbf{i}} + \frac{dr_y(s)}{ds}\hat{\mathbf{j}} + \frac{dr_z(s)}{ds}\hat{\mathbf{k}}. \tag{3.40}$$

Next, $\hat{\mathbf{n}}(s)$ and $\hat{\mathbf{b}}(s)$ can be calculated

$$\hat{\mathbf{n}}(s) = \frac{\frac{d\hat{\mathbf{u}}(s)}{ds}}{\|\frac{d\hat{\mathbf{u}}(s)}{ds}\|} \tag{3.41}$$

$$\hat{\mathbf{b}}(s) = \hat{\mathbf{u}}(s) \times \hat{\mathbf{n}}(s). \tag{3.42}$$

These vectors are illustrated in Figure 3.18.

In the ODE SEK joint the translational motion and reaction spaces can now be defined as

$$\mathcal{M}_T = \hat{\mathbf{u}}(s), \tag{3.43}$$

$$\mathcal{F}_T = \langle \hat{\mathbf{n}}(s), \hat{\mathbf{b}}(s) \rangle. \tag{3.44}$$

The rotational motion and reaction spaces are the same as for a prismatic joint as shown in Equations (3.38) and (3.39).

The relative velocity ($\vec{v}_{J/I}$) and acceleration ($\vec{a}_{J/I}$) between the two constrained bodies

enforced by the joint must be defined. Using the identity

$$\dot{\vec{A}} = \frac{d\vec{A}}{dt} + \vec{\Omega} \times \vec{A} \tag{3.45}$$

where $\vec{A}$ is the vector to be differentiated and $\vec{\Omega}$ is the angular velocity of the frame in which $\vec{A}$ is expressed [102], $\vec{v}_{J/I}$ can be computed as the time derivative of $\vec{r}_{J/I}(s)$ from (3.34)

$$\vec{v}_{J/I} = \dot{\vec{r}}_{J/I}(s) = \dot{s}\hat{\mathbf{u}}(s) + \vec{\omega}_I \times \vec{r}_{J/I}(s). \tag{3.46}$$

Differentiating again gives the relative acceleration

$$\begin{aligned} \vec{a}_{J/I} = \dot{\vec{v}}_{J/I} &= \dot{\vec{\omega}}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times \dot{\vec{r}}_{J/I}(s) + \ddot{s}\hat{\mathbf{u}} + \dot{s}\dot{\hat{\mathbf{u}}} \\ &= \dot{\vec{\omega}}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times (\vec{\omega}_I \times \vec{r}_{J/I}(s) + \dot{s}\hat{\mathbf{u}}) + \ddot{s}\hat{\mathbf{u}} + \dot{s}\dot{\hat{\mathbf{u}}} \\ &= \dot{\vec{\omega}}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times \vec{\omega}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times \dot{s}\hat{\mathbf{u}} + \ddot{s}\hat{\mathbf{u}} + \dot{s}\dot{\hat{\mathbf{u}}} \end{aligned} \tag{3.47}$$

where $\vec{\omega}_I$ is the angular velocity of body I, and $\dot{\hat{\mathbf{u}}}$ is the time derivative of (3.40).

Next, the rotation kinematics are defined. The approach is a straight forward implementation of the rigid body kinematics found in [102]. Body-fixed Euler angles are used, and for the purposes of nomenclature in this section a body fixed 3-1-3 rotation scheme shall be used, although the implementation of the joint allows for any valid Euler angle sequence to be used. Given the Euler angle functions defined in (3.35), rotation matrices that define the orientation of body J with respect to body I can be assembled:

$$[R_1] = \begin{bmatrix} \cos(\theta(s)) & \sin(\theta(s)) & 0 \\ -\sin(\theta(s)) & \cos(\theta(s)) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.48}$$

$$[R_2] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi(s)) & \sin(\phi(s)) \\ 0 & -\sin(\phi(s)) & \cos(\phi(s)) \end{bmatrix}, \tag{3.49}$$

$$[R_3] = \begin{bmatrix} \cos(\psi(s)) & \sin(\psi(s)) & 0 \\ -\sin(\psi(s)) & \cos(\psi(s)) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.50}$$

and

$$[R_{J/I}] = [R_3][R_2][R_1]. \tag{3.51}$$

At any point along the reference path, the relative orientation of the bodies can be determined using the matrix $[R_{J/I}]$.

The relative angular velocity at a given position is determined using the time derivative of each angle function and the appropriate transformation, such that

$$\vec{\omega}_{J/I} = \frac{d\theta(s)}{dt}\hat{\mathbf{k}}_I + \frac{d\phi(s)}{dt}\hat{\mathbf{i}}_1 + \frac{d\psi(s)}{dt}\hat{\mathbf{k}}_2 \tag{3.52}$$

where $\hat{\mathbf{k}}_I$ is the z-axis of the frame on body I, $\hat{\mathbf{i}}_1$ is the x-axis of the frame on body I with $[R_1]$ applied, and $\hat{\mathbf{k}}_2$ is the z-axis of the frame on body I with $([R_2][R_1])$ applied.

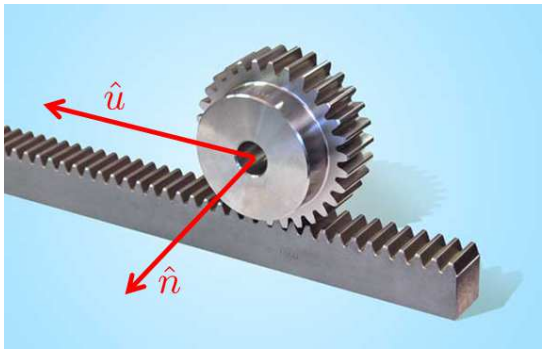
The angular acceleration is

$$\vec{\alpha}_{J/I} = \frac{d\vec{\omega}}{dt} \tag{3.53}$$

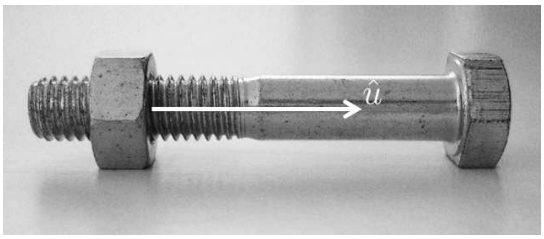
by definition.

Similar to the discussion in Section 3.1, to ensure the dynamics of the joint are correct, the rotational dynamics must be coupled to the translation dynamics. This is required because the joint is defined as having a single translational DOF, and so given the current definition of the joint the rotational dynamics will not appear in the equations of motion. Consider the rack and pinion gear, and nut and bolt systems shown in Figure 3.19. In both cases, if friction is ignored and the systems are considered ideal, a force applied along the motion space ($\hat{\mathbf{u}}$) will cause acceleration; alternatively a torque applied about the appropriate axis ($\hat{\mathbf{n}}$ in Figure 3.19a, and $\hat{\mathbf{u}}$ in Figure 3.19b) can have the same effect. Without additions to the formulation of the SEK joint, any torques will be absorbed as joint reactions because the motion space for rotations is empty, as shown in Equation (3.38).

The approach to couple the two domains follows a similar approach to that used for the DAE SEK joint in Section 3.1. The goal is to transform the required rotation dynamics to the translational domain by applying an equivalent force acting along the translational motion space ($\hat{\mathbf{u}}$). This is done using an approach inspired by generalized forces in Lagrangian mechanics. A generalized force (Q_j) is used to represent the virtual work done by all forces acting on a body along a given generalized coordinate (q_j). For a given generalized



(a) Rack and pinion system with translation motion space shown



(b) Nut and bolt system with translation motion space shown

Figure 3.19: Examples of simple physical systems with coupled translations and rotations

coordinate, the generalized force is given by

$$Q_j = \sum \vec{F} \cdot \frac{\partial \vec{r}}{\partial q_j} \tag{3.54}$$

where $\sum \vec{F}$ is the sum of all forces acting on the body and $\frac{\partial \vec{r}}{\partial q_j}$ is the change in position ($\vec{r}$) for a given change in the generalized coordinate (q_j) [102].

In the ODE SEK joint this concept can be used to project all joint reaction torques onto the translational motion space which will ensure that the correct coupling between the translation and rotation dynamics is enforced. Replacing some of the symbols, (3.54) can be rewritten as

$$F_{TP} = \vec{T}_{net} \cdot \vec{p}(s) \tag{3.55}$$

which gives the torque projection force, F_{TP}. In (3.55), $\vec{T}_{net}$ is the reaction torque in the joint and $\vec{p}(s)$ is the change in the relative orientation of the two constrained bodies with respect to s. Given the three Euler angle rotations from (3.35), $\vec{p}(s)$ is defined as

$$\vec{p}(s) = \frac{\partial \theta(s)}{\partial s}\hat{\mathbf{k}}_I + \frac{\partial \phi(s)}{\partial s}\hat{\mathbf{i}}_1 + \frac{\partial \psi(s)}{\partial s}\hat{\mathbf{k}}_2 \tag{3.56}$$

which gives the change in body orientation with respect to the displacement along the motion space. F_{TP} is then applied along $\hat{\mathbf{u}}$ which ensures the rotational dynamics of the joint are correctly represented.

For further explanation, consider a simple system of a disc of mass m, radius r and

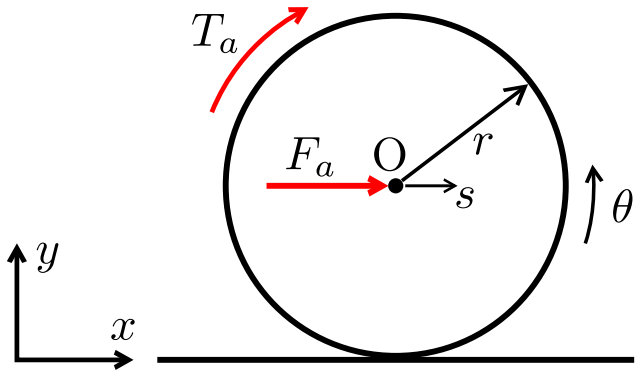


Figure 3.20: Problem description for simple torque projection example problem

inertia I rolling without slipping, being driven by an applied force and moment, as shown in Figure 3.20. The kinematic relationship between translation and rotation is

$$\theta = -\frac{s}{r} \tag{3.57}$$

and

$$\frac{\partial \theta}{\partial s} = -\frac{1}{r}. \tag{3.58}$$

The free body diagram of the system is shown in Figure 3.21. Summing the forces in the x-direction gives:

$$\sum F_x : \; F_a - F_R = m\ddot{s}. \tag{3.59}$$

Summing the moments about point O and using the kinematic relationship shown in (3.57) gives:

$$\begin{aligned} \sum M_O : \; -T_a - rF_R &= I\alpha \\ &= -I\ddot{\theta} \\ &= -I\frac{\ddot{s}}{r}. \end{aligned} \tag{3.60}$$

Combining (3.59) and (3.60) gives the equation of motion for the system:

$$\left(\frac{I}{r} + rm\right)\ddot{s} - rF_a - T_a = 0. \tag{3.61}$$

Referring once again to the free body diagram in Figure 3.21 and using a formulation

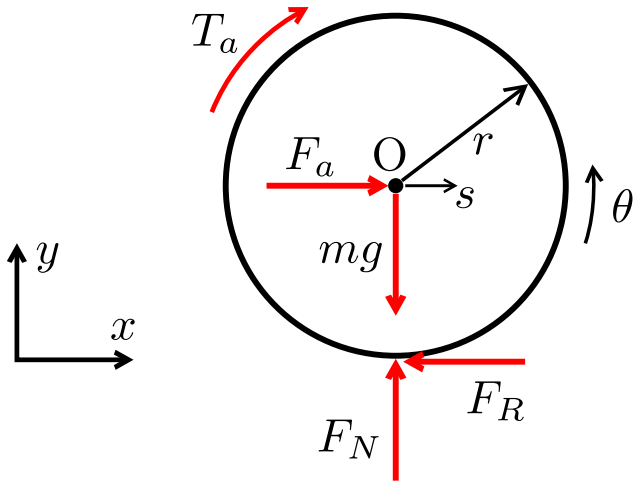


Figure 3.21: Free body diagram for simple torque projection example problem

that follows the definition of the ODE SEK joint, since there is only 1 DOF, in this case along the x-axis, the single equation of motion obtained will be

$$m\ddot{s} = F_a - F_R + F_{TP} \tag{3.62}$$

where F_{TP} is the torque projection force that is calculated using (3.55). First, $\vec{p}$ is calculated; for a rolling disc with the kinematic relationship described in (3.57) this becomes

$$\vec{p} = \frac{\partial\theta}{\partial s}\hat{\mathbf{k}} = -\frac{1}{r}\hat{\mathbf{k}}. \tag{3.63}$$

To compute $\sum \vec{T}_{net}$ all reaction torques must be considered, remembering that the rotation motion space is empty, and therefore all externally applied torques will generate equal and opposite reaction torques. These torques are shown in Figure 3.22.

$$\vec{T}_{net} = (-rF_R - T_a + I\alpha)\hat{\mathbf{k}} \tag{3.64}$$

which means that

$$\begin{aligned} F_{TP} = \vec{T}_{net} \cdot \vec{p} &= (-rF_R - T_a + I\alpha)\hat{\mathbf{k}} \cdot \left(-\frac{1}{r}\right)\hat{\mathbf{k}} \\ &= F_R + \frac{T_a}{r} - I\frac{\ddot{s}}{r^2} \end{aligned} \tag{3.65}$$

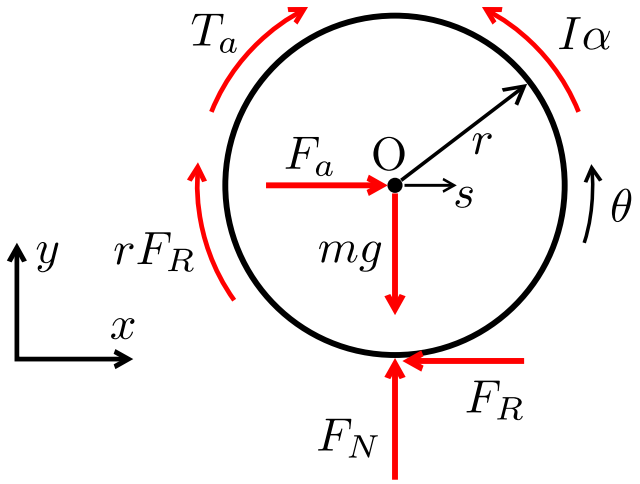


Figure 3.22: ODE SEK formulation explanation for simple torque projection example problem

which gives F_{TP}; the coupling force that should be applied along the motion space. Combining (3.62) and (3.65) gives

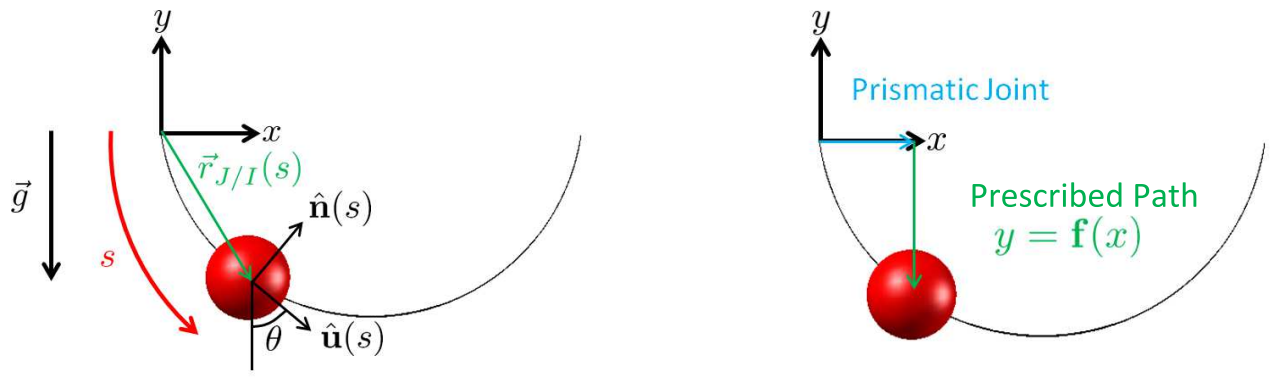
$$\left(\frac{I}{r} + rm\right)\ddot{s} - rF_a - T_a = 0 \tag{3.66}$$

which is the same as (3.61). The example gives a very simple but practical example of the theory of the torque projection used in the ODE SEK joint. This coupling ensures that the translational and rotational dynamics in the ODE SEK joint are coupled. As shown in the example, without the coupling the dynamic equation for the system is not correct.

With the torque projection added to the joint formulation, the theory for the ODE SEK joint is complete. The translation and rotation kinematics are described as functions of a single coordinate, the path length, which eliminates all algebraic dependencies. The coupling of the translation and rotation dynamics is accomplished using a torque projection approach inspired by generalized forces from Lagrangian mechanics.

Example: semi-circular path

A simple example using a semi-circular path is now shown. A circular path is chosen because a simple analytical solution for the position vector in terms of the path-length exists.



(a) ODE joint

(b) DAE joint

Figure 3.23: Kinematics of planar particle ODE SEK joint with semi-circular path

Consider a particle of mass m constrained to move along a planar semi-circular path of unit radius in the x-y plane and subject to a gravitational force in the negative y direction, as shown in Figure 3.23. The displacement vector across the joint is

$$\vec{r}_{J/I}(s) = (-\cos(s) + 1)\hat{\mathbf{i}} - \sin(s)\hat{\mathbf{j}}. \tag{3.67}$$

Using (3.40), $\mathcal{M}_T$ can be computed:

$$\hat{\mathbf{u}}(s) = \sin(s)\hat{\mathbf{i}} - \cos(s)\hat{\mathbf{j}}. \tag{3.68}$$

$\mathcal{F}_T$ is given by:

$$\langle \hat{\mathbf{n}}, \hat{\mathbf{b}} \rangle = \langle \cos(s)\hat{\mathbf{i}} + \sin(s)\hat{\mathbf{j}}, \hat{\mathbf{k}} \rangle. \tag{3.69}$$

Figure 3.23a shows the implementation of the constraint. Of note, compared with the DAE SEK joint presented in Section 3.1 and shown in Figure 3.23b, is that two dependent coordinates x and y have been replaced with s in the reformulated ODE joint.

For the simple system shown, the single dynamic equation governing the dynamics of the particle is

$$m\ddot{s} - mg\cos(s) = 0. \tag{3.70}$$

3.2.2 Path generation

Unfortunately other shapes cannot always be described in terms of path length using analytical expressions and so a numerical path generation scheme is required. In this thesis, a path generation algorithm has been created that can accurately generate mathematical functions describing a given path as a function of its path length. This work is carried out as a pre-processing step, to ensure the mathematical complexity (and therefore simulation speed) of the final joint formulation is not negatively impacted. The concept of this algorithm is to use splines to represent data that has been parameterized in terms of its path length. Before discussing the steps in the algorithm, some background information is provided. First, the path length calculation is explained and then some details on the two spline fitting algorithms that have been used are presented. Finally, everything is assembled into the final path fitting algorithm that converts a users input numerical data into the spline functions required by the SEK joint.

Path length calculation

Given the spatial path defined by $y = f(x)$, $z = g(x)$, the length of the path from x_0 to point a can be calculated using [104]:

$$s = \int_{x_0}^{a} \sqrt{1 + \left(\frac{df(x)}{dx}\right)^2 + \left(\frac{dg(x)}{dx}\right)^2}\,\mathrm{d}x. \tag{3.71}$$

Given the parameterized spatial path defined by $x = f(\tau)$, $y = g(\tau)$, $z = h(\tau)$, the length of the path from τ_0 to $\tau = a$ can be calculated using:

$$s = \int_{\tau_0}^{a} \sqrt{\left(\frac{df(\tau)}{d\tau}\right)^2 + \left(\frac{dg(\tau)}{d\tau}\right)^2 + \left(\frac{dh(\tau)}{d\tau}\right)^2}\,\mathrm{d}\tau. \tag{3.72}$$

Even for simple shapes, there is no closed form solution to compute the path length of an arbitrarily shaped curve. Consider a parabola described by $y = x^2$; using (3.71) the integral becomes:

$$s = \int_{x_0}^{a} \sqrt{1 + 4x^2}\,\mathrm{d}x \tag{3.73}$$

which has no closed form solution. Clearly an analytical approach is not a practical solution for a generic path generation algorithm.

Spline fitting

Two different types of splines are used in the path generation algorithm. When fitting splines to experimentally obtained data, the adaptive knot placement algorithm developed by Tao et al. is used [91] to create accurate B-splines. This algorithm helps to find the optimal location of the minimal number of internal knots, k, and was described in detail in Section 2.4. The algorithm iteratively increases the number of knots in the spline and is implemented in MATLAB. After each knot is added, MATLAB's `fmincon` function with the `interior-point` algorithm [105] is used to determine the knot locations that minimize the error between the spline curve and the user input table data to which the spline is being fit.

When fitting splines to high-resolution, clean data where over-fitting the splines to experimental noise is not a concern, a new knot fitting algorithm has been developed. This method, called the "iterative knot finding algorithm" has been found to give comparable results with much less processing time than the Tao method.

Conceptually the "iterative knot finding algorithm" is a sequential search for segments of the data that can be represented by polynomials of a specified order. First, the data is segmented, such that each segment has at least $k + 1$ points, where k is the order of the polynomial to be fit. The resolution of this segmentation defines the maximum number of knots that can be in the spline, and is used as a tuning parameter for the algorithm. Next, starting at the beginning of the curve, the algorithm iterates along the segments of the curve, sequentially fitting a polynomial to the data in the current segment, increasing the size of the segment and refitting the polynomial to the larger data set. If the fit error of the second polynomial is larger than the fit error of the initial polynomial, beyond a user-defined threshold (also used as a tuning parameter), then the location separating the two segments becomes a knot location, and the process starts again. If the fit error between the two polynomials is lower than the threshold, then the process of increasing the segment continues until either the end of the data is reached or the error difference threshold is exceeded. This error threshold is a second tuning parameter that can be used to affect the behaviour of the algorithm. The pseudo-code for the iterative knot finding approach is shown in Algorithm 1.

Because of the way the segments are created and because there is no modification of the segmentation bounds during the running of the algorithm, the resulting knot locations

Algorithm 1: Iterative knot finding algorithm

input : data (s,x)
polynomial order, k
max number of knots
error threshold
output: knots[]: an array of optimal knot locations

knots = [];
i = 0;
$s_{end_{curr}}$ = s(1);

while $s_{end_{curr}}$ = *s(end)* **do**
  s_{start} = $s_{end_{curr}}$;
  p = polynomial of order k over interval [s_{start}, s_{start} + increment];
  current error = error of fit(p);
  counter = 2;
  repeat
    previous error = current error;
    p = polynomial of order k over interval [s_{start}, s_{start} + counter ∗ increment];
    current error = error of fit(p);
    counter++;
  until *(current error - previous error) > threshold*;
end

are not necessarily the globally optimal solution. However, if appropriate values for the tuning parameters are used, the results are near-optimal and have performed adequately for the needs of this work.

Path generation algorithm

An overview of the path generation algorithm is shown in Figure 3.24. All of the logic has been implemented in MATLAB. The goal of the path generation algorithm is to take numerical data defining a spatial curve as an input, and output symbolic functions describing the curve in terms of the path length. The path length parameterization is important for the computation of the path variables as discussed in Section 3.2.1. The functions used to define the Euler angles for the body orientation are also generated as part of this process.

The first step is to parameterize the numerical table data input by the user in terms of the path length. To compute the path length, the numerical data must be differentiated.

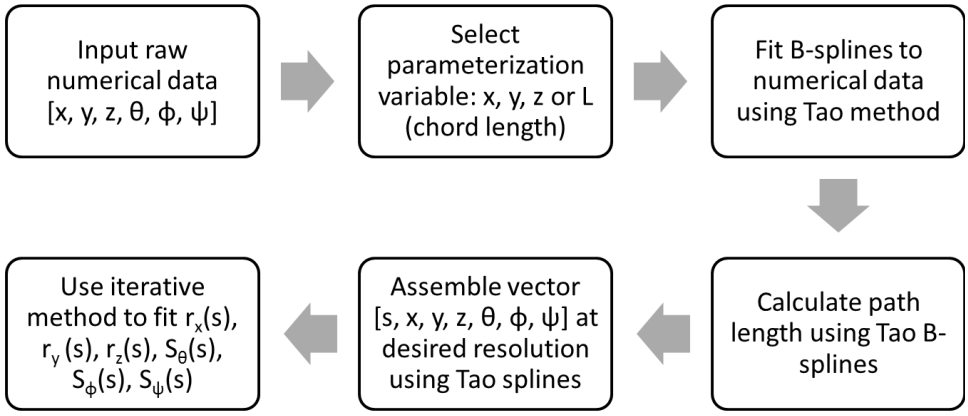


Figure 3.24: Overview of the path generation process

However, it is well known that numerical differentiation can amplify any inaccuracies or noise in the data. To avoid numerical differentiation, splines are fit directly to the data. This approach has been suggested by Wang et al. [104]. It is most straightforward to define the spline as a function of x, y, or z. However this is not possible if the data does not represent a true function, i.e. the independent values do not map to unique dependent values, such as with a planar circle. In this case the data may need to be parameterized before the initial splines can be fit. In situations like this, the chord length (L) between each data point is computed and is used as the independent variable instead of x, y, or z. Because the chord length is simply the distance between neighbouring data points it can be very easily calculated directly from the numerical data using (3.74), and this is why it is chosen as the alternate parameterization variable if it is not possible to use x, y, or z. Figure 3.25 shows the difference between path length and chord length.

$$L = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \tag{3.74}$$

Once the parameterization has been chosen, the Tao knot finding method is used to generate the initial splines. The Tao method is used here because of its ability to fit noisy experimental data, and the stopping conditions are designed to prevent over-fitting. Once the initial splines are fit, the path length can be calculated by differentiating the splines and integrating, using (3.71) and (3.72). Using these splines, high-resolution data for the path length, position components and three Euler angles is generated.

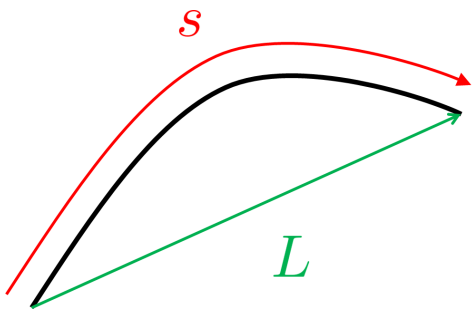
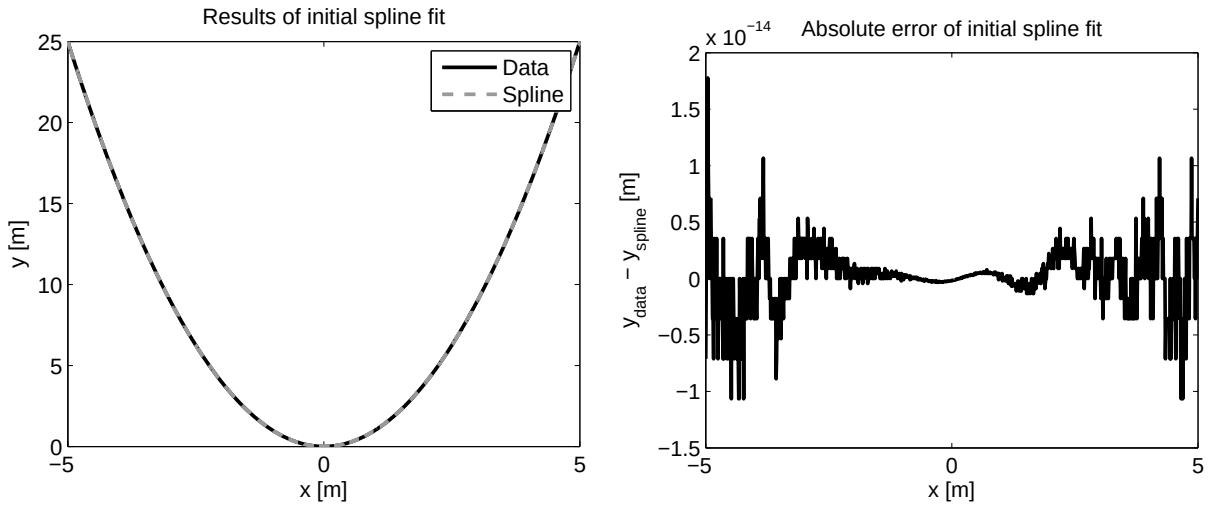


Figure 3.25: Definition of path length (s) and chord length (L)

Next, using the high-resolution data, the iterative knot finding method is used to fit the final path length parameterized position splines $(r_x(s), r_y(s), r_z(s))$, for use in the ODE SEK joint definition, as shown in (3.34). According to (3.40), differentiation of the position B-splines gives the tangential unit vector ($\hat{\mathbf{u}}$). This symbolic derivative is used directly in the joint definition. From (3.41), the derivative of the tangential unit vector can be used to find the normal unit vector. Because the magnitude must be normalized to unity it is not possible to simply use the symbolic derivative of the tangential unit vector. Instead, the splines representing $\frac{d\hat{u}}{ds}$ are used to generate numerical data along the given path, that is then normalized, and the iterative spline fitting algorithm is used again to generate the splines for the normal unit vector. This is done in the pre-processing step to ensure that the magnitude of the spline does not need to be computed symbolically by Maple. The binormal unit vector, however is not determined in the pre-processing, and a symbolic cross product (3.42) is used in the joint definition implemented in MapleSim.

Once the B-splines for the position vector and tangential and normal unit vectors have been generated, the splines describing the rotation kinematics are created. The final step is to convert the splines to a form that can easily be used in the ODE SEK joint definition. Having used the flexibility and extensive research literature centered around B-splines to create accurate interpolating functions for each of the curves, the B-splines are converted to a simple piecewise polynomial representation [61]. This is done because the piecewise polynomials are a simpler mathematical definition of the spline when compared to B-splines. Schumaker suggests that if a B-spline is to be evaluated at least twice, then it is more time-efficient to convert it to a piecewise polynomial than evaluating the B-spline representation [89]. These piecewise polynomials are then used in the ODE SEK joint definition to define the joint kinematics and the motion and reaction spaces.



(a) Initial spline fit

(b) Error of initial spline fit

Figure 3.26: Results of initial spline fit for planar parabolic path

Parabolic path example

An example using a parabolic path is now shown to demonstrate the work flow of the path generation procedure. For this example the path is a planar parabola, and no rotation splines will be generated; a simple particle will be simulated. The parabola is defined as $y = x^2$, and the first step is to create "experimental" numerical data in the form $[x\ y]$ that will be used as the input for the path generation algorithm. In this simple example x can be used to parameterize the data, and so the first step is to use the Tao knot finding algorithm to fit a quintic B-spline ($y = f(x)$) to the generated numerical data. The results of this are shown in Figure 3.26. The optimal knot locations, found by the Tao method, are shown and it can be seen that the resulting spline fits the input data well, with very small absolute error.

The next step is to numerically compute the path length using (3.71). As the path length is calculated along the curve using Wang's methodology [104] a numerical vector of the form $[s\ x\ y]$ is assembled, to be used in the next step of the process.

Using the numerical data which is now parameterized in terms of path length it is now possible to fit the final splines $r_x(s)$ and $r_y(s)$. This is where the iterative knot finding method is used. Because the position splines are defined as a function of the path length, the tangential unit vector ($\hat{\mathbf{u}}$) can be calculated directly using (3.40). Because $\hat{\mathbf{u}}$ is a unit

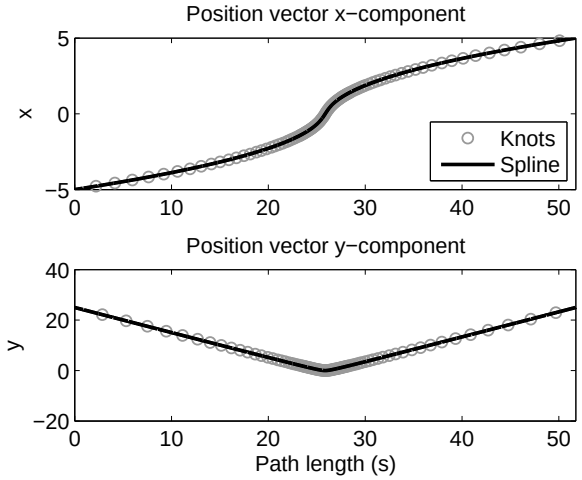


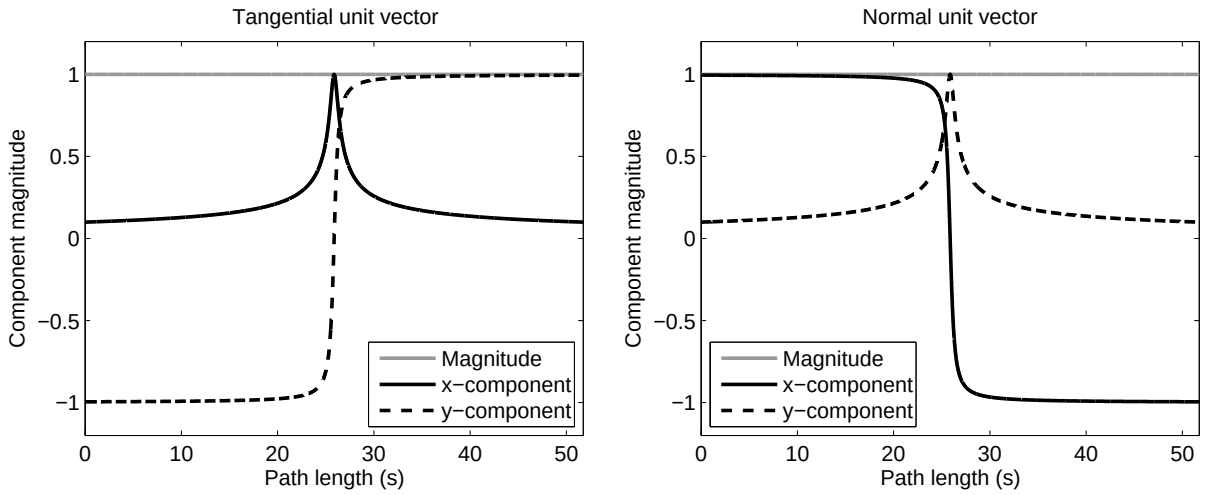
Figure 3.27: Final ODE SEK joint path position vectors representing a planar parabola in terms of path length

vector, its magnitude must be unity, which provides an excellent way to test the combined accuracy of the final position splines:

$$Error = \left\| \sqrt{\left(\frac{dr_x(s)}{ds}\right)^2 + \left(\frac{dr_y(s)}{ds}\right)^2} - 1 \right\|. \tag{3.75}$$

Once the iterative knot finding method is used to find the optimal knots and fit the position and tangential unit vector splines, the normal vector can be calculated. The splines representing the components of each of these vectors can be seen in Figures 3.27 and 3.28. Once the splines are fit, they are converted to piecewise polynomials and printed to a file in Maple syntax to be used directly in the MapleSim implementation of the ODE SEK joint.

As seen in the previous example, even a seemingly simple shape like a parabola can present challenges to parameterize in terms of its path length. The developed algorithm is able to take generic numerical data and generate mathematical functions describing the curve in terms of its path length for use in the ODE SEK joint. More complex applications of this algorithm are shown in Chapter 4.



(a) Tangential unit vector ($\hat{\mathbf{u}}$) (b) Normal unit vector ($\hat{\mathbf{n}}$)

Figure 3.28: Final ODE SEK joint path unit vectors representing a planar parabola in terms of path length

Composite path generation

A problem discussed by Kecskeméthy et al. [66] is that in cases where $\frac{d\hat{\mathbf{u}}}{ds}$ vanishes it is not possible to calculate the normal vector. This happens at segments where the path becomes straight (i.e. $\hat{\mathbf{u}}$ is constant) and the problem is caused by a singularity in (3.41), since the denominator becomes 0. Kecskeméthy et al. proposed a method using limit analysis to handle the cases where $\frac{d\hat{\mathbf{u}}}{ds}$ vanishes, and a similar, albeit simpler, approach has been implemented in the ODE SEK joint path generation code. To ensure the mathematical simplicity of the joint definition is not compromised, all of these calculations have been implemented in the pre-processing path generation code, rather than in the joint formulation. While Kecskeméthy's algorithm identifies the straight segments in the user supplied data, the algorithm implemented for the ODE SEK joint path generation requires that the user manually identify the straight segments in the curve. This approach is similar to that used by Pombo and Ambrósio [57, 58] in which the user can create "curved" or "straight" segments, and group them together to form a composite path. The approach of splitting the curve into multiple segments is desired as splines can exhibit oscillations when transitioning from straight to curved regions of the data [60]. As discussed in Section 2.4 Akima splines [60] and shape preserving splines [62, 63] have been developed to

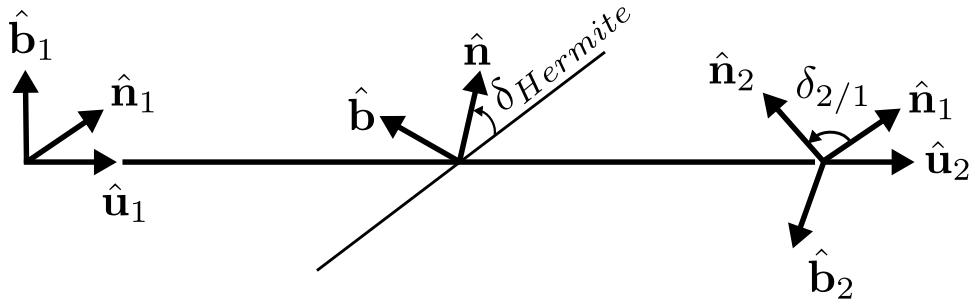


Figure 3.29: Example of variation of the normal vector along a straight segment

overcome the oscillation problem, but each of these spline representation has its own set of disadvantages that make them not ideal for application in the ODE SEK joint.

Once all of the segments have been correctly defined, the fitting algorithm can begin. Each segment is processed in order, starting from the beginning of the curve. The processing of the segment differs depending on whether the segment is curved or straight. The process to handle a curved segment is unchanged from the algorithm described previously. If there are consecutive curved segments, then it is the responsibility of the user to ensure that the data is continuous. For straight segments, once again, it is the responsibility of the user to ensure that the data will allow the tangential unit vectors to be continuous across the transition between neighbouring curved and straight segments. However, it is perfectly reasonable to expect that the normal vector will need to be different at each end of the straight segment to ensure the resulting composite curve for the normal unit vector is smooth and continuous, as shown in Figure 3.29. The normal vector at the start of the straight segment is determined by the final value for the normal vector of the previous segment. Likewise, the normal vector at the end of the straight segment is determined by the initial value for the normal vector of the next segment. Therefore, the normal vector needs to vary along the length of the straight segment to ensure continuity at each end.

Because the segment is straight, the tangential unit vector will remain constant along the length of the segment. The first step is to compute the angle between the two normal vectors at each end of the segment, $\delta_{2/1}$:

$$\delta_{2/1} = \arccos(\hat{\mathbf{n}}_\mathbf{1} \cdot \hat{\mathbf{n}}_\mathbf{2}). \tag{3.76}$$

Next, a Hermite blending function [106] is used to create a smooth blending for the

angle of the normal vector, from δ_1 to δ_2. δ_1 represents the relative angle of the normal vector at the beginning of the straight segment, and has a value of 0. δ_2 represents the relative angle of the normal vector at the end of the of the straight segment, and is equal to $\delta_{2/1}$. δ_{linear} is the linear interpolation in the interval $[\delta_1, \delta_2]$, such that

$$\delta_{linear} = \frac{\delta_{2/1}}{L} s \tag{3.77}$$

where L is the length of the straight segment, and s is the distance along the straight segment. δ_{linear} can be mapped to a blended value, $\delta_{Hermite}$, using:

$$\delta_{Hermite} = \delta_2 \left\{ -2 \left(\frac{\delta_{linear} - \delta_1}{\delta_2 - \delta_1} \right)^3 + 3 \left(\frac{\delta_{linear} - \delta_1}{\delta_2 - \delta_1} \right)^2 \right\} \tag{3.78}$$

which ensures smoothness along the straight segment and at the transitions between the neighbouring segments.

The normal vector at any point along the straight segment is $\hat{\mathbf{n}}_\mathbf{1}$ rotated by the angle $\delta_{Hermite}$ about $\hat{\mathbf{u}}$. To compute this normal vector, $\hat{\mathbf{n}}$, a rotation matrix is required. Euler's Theorem allows the computation of a rotation matrix, $[R_\delta]$, about an arbitrary axis, in this case $\hat{\mathbf{u}}$ [107]

$$[R_{\delta_{Hermite}}] = \cos(\delta_{Hermite})[1] - \sin(\delta_{Hermite})[\widetilde{u}] + (1 - \cos(\delta_{Hermite}))\{\hat{\mathbf{u}}\}\{\hat{\mathbf{u}}\}^T. \tag{3.79}$$

where $[\widetilde{u}]$ is the skew symmetric matrix of $\hat{\mathbf{u}}$ and [1] is the identity matrix. The normal vector at any point along the straight segment can now be computed using

$$\{\hat{\mathbf{n}}\} = [R_{\delta_{Hermite}}] \{\hat{\mathbf{n}}_\mathbf{1}\}. \tag{3.80}$$

Splines for each component of the normal vector along the straight segment can now be computed using the methods discussed previously. The path length of a straight line is a simple concept, and the iterative knot finding algorithm can be used to generate a spline for each component of $\hat{\mathbf{n}}$.

Once the normal vector splines for all straight segments have been computed, the final step is to assemble the piecewise spline functions from each segment into a single piecewise spline function that describes the full composite curve. These are then exported from Matlab into MapleSim syntax for use in the ODE SEK joint. An example roller coaster

model has been simulated using this path generation strategy and the full example is shown in Section 4.3.

3.2.3 Validation

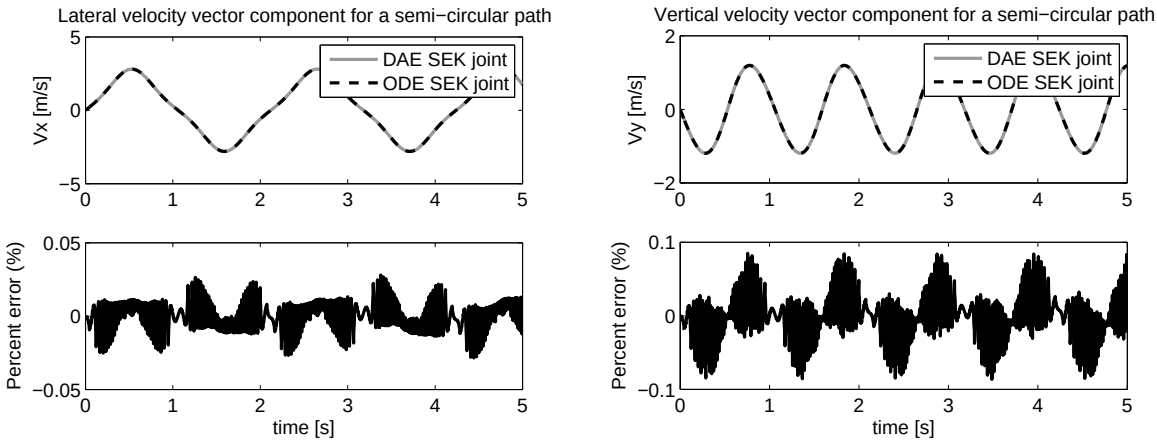
To ensure the joint accurately represents the physical behaviour of a 1-DOF system, validation is required. The first validation uses the particle moving along a planar semi-circular path that was discussed previously, and shown in Figure 3.23a. The mass, m of the particle is set to 1kg, and the system is modeled using the DAE and new ODE formulations of the joint. Since the DAE formulation of the joint was extensively validated against Adams/View, using it as a basis for comparison with the new formulation is fair. The particle is allowed to fall along the semi-circular path under the influence of gravity starting from rest at $s = 0.642$m in the ODE formulation or $x = 0.2$m and $y = -0.598$m in the previous DAE formulation. Figure 3.30 shows a comparison between the x and y-component velocities of the DAE and ODE implementations of the joint. In both cases the Maple RKF45 solver is used with integration tolerances of $1e-8$. As in Section 3.1, the system energy is measured to ensure the law of conservation of energy is not violated. In this case, because gravity is present, the system energy is calculated as

$$E = \frac{1}{2} m_{body} v^2 + mgy \tag{3.81}$$

where g is the gravitational acceleration constant, and y is the position along the vertical y-axis. Figure 3.31 shows that the total system energy remains constant throughout the motion, as expected.

To validate the new formulation of the ODE SEK joint using a path from the path generation algorithm, the parabolic path generated in Section 3.2.2 is used. A particle of 1kg falling under the force of gravity is constrained to move along the parabola $y = x^2$. The initial conditions for the simulation are $x = -2.279$m and $y = 5.194$m, and the particle starts from rest. In both cases the Maple RKF45 solver is used with tolerance $1e-8$. The components of the velocity vector are shown, in Figure 3.32, to match those resulting from the simulation of the DAE SEK joint. The small error shown in the plots is due to numerical inaccuracies.

A final validation example is carried out using the same spatial four-bar mechanism that was used for the validation of the DAE version of the joint in Section 3.1.4. The results



(a) Lateral velocity for a semi-circular path

(b) Vertical velocity for a semi-circular path

Figure 3.30: Results of ODE SEK joint simulation using planar semi-circular path

Table 3.6: Simulation time comparison between DAE and ODE SEK joints for spatial four-bar simulation

	DAE Joint	ODE Joint	Improvement
Simulation time [s]	3.853	2.886	1.34x

for the ODE SEK joint were compared with the results from the Adams/View simulation and as shown in Figure 3.33, the results match well. In contrast to the results for the DAE joint, there is very little noise present in the reaction forces and moments thanks to the full symbolic representation of the joint, which means the path is at least C^2-continuous, as opposed to the lookup tables used in the DAE SEK joint which were only C^1-continuous.

To demonstrate the simulation speed improvement offered by the ODE SEK joint compared with the DAE SEK joint, the simulation times are compared between the two joints using the spatial 4-bar example. For the 4 second simulation, the solver was changed to a fixed-step Euler solver with a step size of $1e-5$ seconds; the results in Table 3.6 show a speedup of 25% over the DAE version of the SEK joint. Since Adams/View does not offer a fixed-step solver, and because of the simplicity of the spatial four-bar model, the speed improvements the ODE SEK joint can offer over conventional Adams models will be demonstrated in Chapter 4.

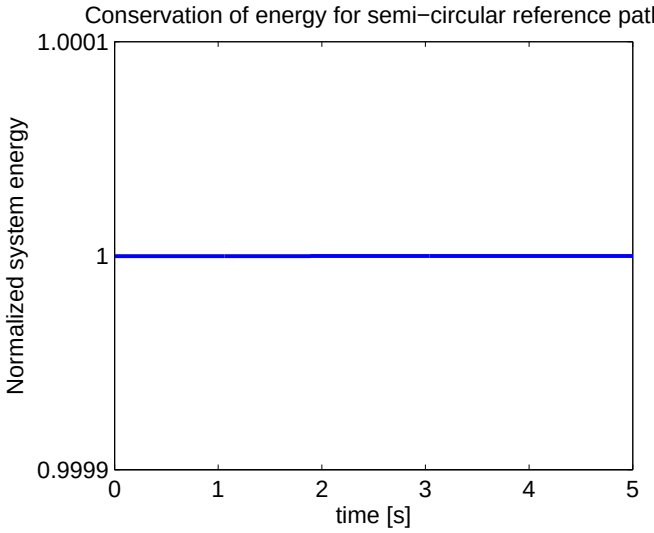
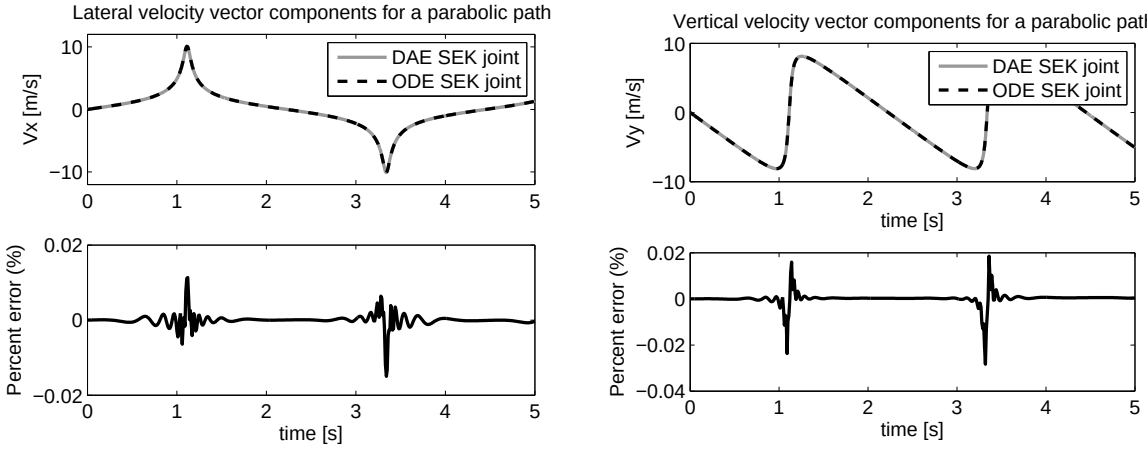


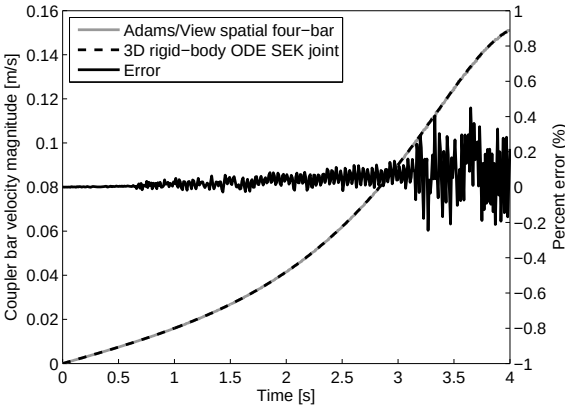
Figure 3.31: Conservation of energy in the ODE SEK joint using a planar semi-circular path



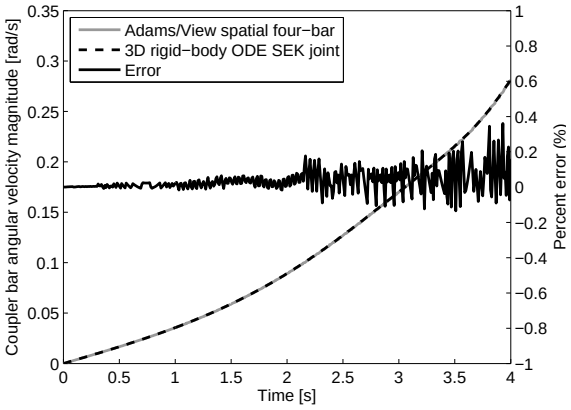
(a) Lateral velocity for a parabolic path

(b) Vertical velocity for a parabolic path

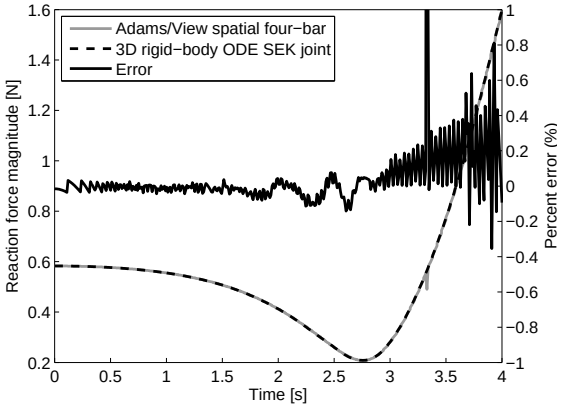
Figure 3.32: Validation of ODE SEK joint for a parabolic path against DAE SEK joint



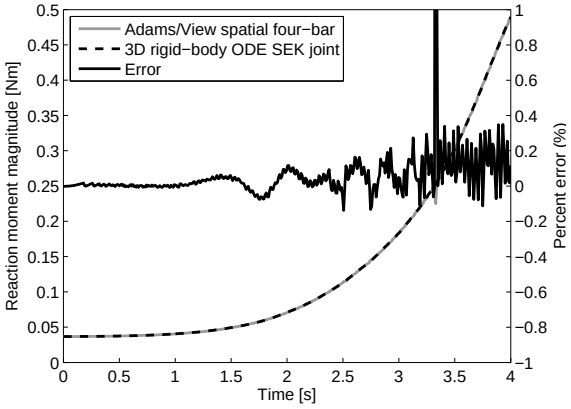
(a) Speed versus time



(b) Angular speed versus time



(c) Reaction force magnitude versus time



(d) Reaction moment magnitude versus time

Figure 3.33: Validation of ODE SEK joint against Adams/View spatial four-bar

3.2.4 Compliant SEK joint

As discussed in Section 2.3 it is important to consider system compliance in many cases for which the SEK joint may be applied. In this section an extension of the ODE SEK joint to introduce compliance is discussed. To introduce compliance to the ODE SEK joint, the DOF is increased from one to six. The five additional coordinates that represent the small displacements about the specified path are:

1. n, the translational deflection in the $\hat{\mathbf{n}}$ direction
2. b, the translational deflection in the $\hat{\mathbf{b}}$ direction
3. θ_d, the rotational deflection about the first body-fixed Euler rotation axis
4. ϕ_d, the rotational deflection about the second body-fixed Euler rotation axis
5. ψ_d, the rotational deflection about the third body-fixed Euler rotation axis

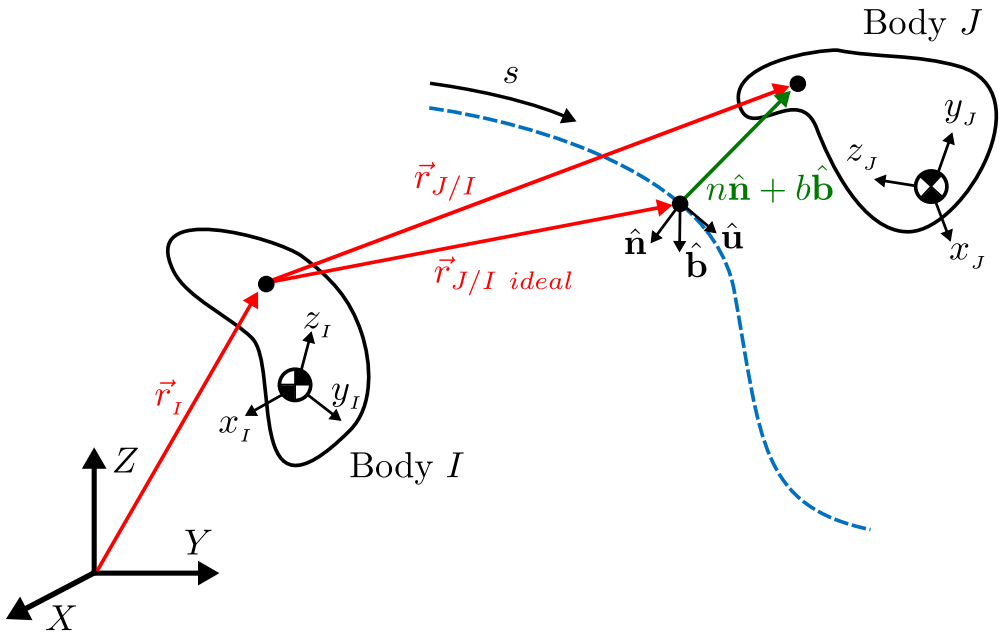


Figure 3.34: General kinematics of compliant ODE SEK joint

The position vector of the rigid or ideal ODE SEK joint, $\vec{r}_{J/I}$, from (3.34) is renamed to $\vec{r}_{J/I\ ideal}$:

$$\vec{r}_{J/I\ ideal}(s) = r_x(s)\hat{\mathbf{i}} + r_y(s)\hat{\mathbf{j}} + r_z(s)\hat{\mathbf{k}} \tag{3.82}$$

The two additional translation coordinates, n and b, are incorporated into a revised joint displacement vector, $\vec{r}_{J/I}$, in which the displacements due to deflections are considered:

$$\vec{r}_{J/I} = \vec{r}_{J/I\ ideal}(s) + n\hat{\mathbf{n}} + b\hat{\mathbf{b}}. \tag{3.83}$$

This is illustrated in Figure 3.34. Following a similar derivation process to that used for the rigid SEK joint in Equations (3.46) and (3.47), the equations for the velocity and acceleration across the joint become:

$$\vec{v}_{J/I} = \vec{\omega}_I \times \vec{r}_{J/I}(s) + \dot{s}\hat{\mathbf{u}} + \dot{n}\hat{\mathbf{n}} + \dot{b}\hat{\mathbf{b}}. \tag{3.84}$$

$$\vec{a}_{J/I} = \dot{\vec{\omega}}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times \vec{\omega}_I \times \vec{r}_{J/I}(s) + \vec{\omega}_I \times \dot{s}\hat{\mathbf{u}} + \ddot{s}\hat{\mathbf{u}} + \dot{s}\dot{\hat{\mathbf{u}}} + \ddot{n}\hat{\mathbf{n}} + \dot{n}\dot{\hat{\mathbf{n}}} + \ddot{b}\hat{\mathbf{b}} + \dot{b}\dot{\hat{\mathbf{b}}} \tag{3.85}$$

Similarly, the joint orientations from (3.35) are rewritten to include the new coordinates that represent the rotational deflections:

$$\{\theta(s), \phi(s), \psi(s)\} = \{S_\theta(s) + \theta_d, S_\phi(s) + \phi_d, S_\psi(s) + \psi_d\}. \tag{3.86}$$

The equations for the rotation matrices, angular velocity and angular acceleration remain as originally shown in (3.48) - (3.53).

The motion and reaction spaces are also rewritten as:

$$\mathcal{M}_T = \langle \hat{u}, \hat{n}, \hat{b} \rangle \tag{3.87}$$

$$\mathcal{F}_T = \emptyset \tag{3.88}$$

$$\mathcal{M}_R = \langle \hat{u}, \hat{n}, \hat{b} \rangle \tag{3.89}$$

$$\mathcal{F}_R = \emptyset \tag{3.90}$$

which gives the joint six DOF. The five deflection coordinates are used with spring and damping components to represent a linear bushing model [75]. Two parameters are introduced for each of the new coordinates; a spring stiffness, $k_\square$, and a damping coefficient, $c_\square$, where $\square$ is replaced with each of the coordinates. The forces are defined as

$$F_n = k_n n + c_n \dot{n} \tag{3.91}$$

$$F_b = k_b b + c_b \dot{b} \tag{3.92}$$

and the bushing moments are defined as

$$M_\theta = k_\theta \theta_d + c_\theta \dot{\theta}_d \tag{3.93}$$

$$M_\phi = k_\phi \phi_d + c_\phi \dot{\phi}_d \tag{3.94}$$

$$M_\psi = k_\psi \psi_d + c_\psi \dot{\psi}_d. \tag{3.95}$$

The coupling of the translational and rotational domains is unchanged and so the total force applied across the joint is

$$\vec{F} = F_{TP}\hat{\mathbf{u}} + F_n \hat{\mathbf{n}} + F_b \hat{\mathbf{b}} \tag{3.96}$$

and the total moment applied across the joint is

$$\vec{M} = M_\theta \hat{\mathbf{k}} + M_\phi \hat{\mathbf{i}}' + M_\psi \hat{\mathbf{k}}''. \tag{3.97}$$

Because no similar joints exist in other software available it is not possible to do simple validation examples in this section, and so a validation is carried out using a compliant MacPherson suspension model in Section 4.1.2.

3.3 DEK joint

Another logical extension of the single-DOF SEK joint is to add an additional translational degree of freedom, so that the motion space is now defined by a surface rather than a spatial curve. This results in a two-DOF joint that is called the Double-DOF Equivalent Kinematic (DEK) joint. In a vehicle dynamics scenario this is useful to represent a steered suspension where the wheel moves on a surface. Figure 3.35 shows the camber angle kinematics of a vehicle suspension. The camber angle is a function of both the steering angle and the vertical displacement of the wheel relative to the chassis. The wheel moves along this surface as the suspension is actuated and the driver moves the steering wheel. This is a very fundamental application of the DEK joint. In a biomechanics example it is useful to model the thumb joint [108, 109] or shoulder joint [110, 111].

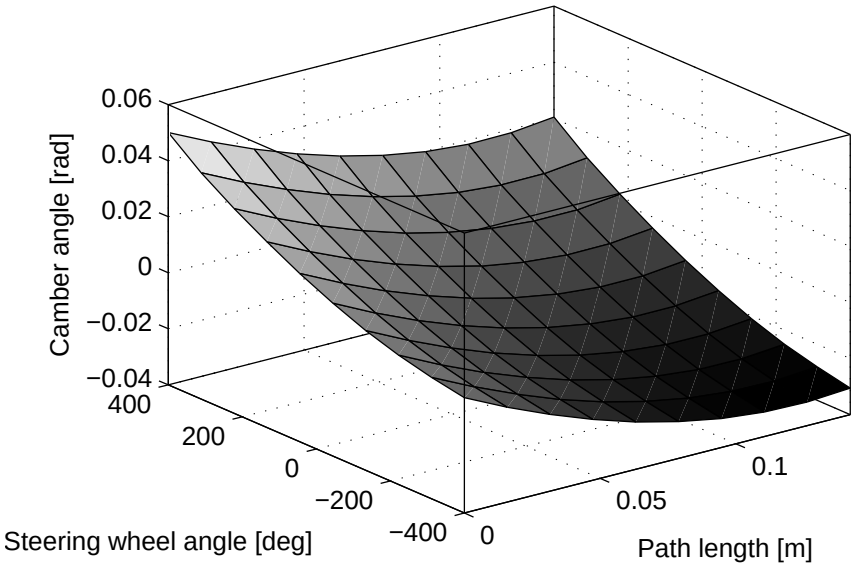


Figure 3.35: Camber kinematics of a steered suspension. The camber angle is a function of both the steering angle and the vertical displacement of the wheel relative to the chassis.

3.3.1 Joint theory

The definition of the DEK joint is very similar to the SEK joint; however unlike the SEK joint, there is no simple way to define the functions in terms of path length, and so any coordinates can be used. This means that an additional calculation is required to ensure that the magnitudes of the two tangential unit vectors are always unity. The two coordinates used will be represented as s_1 and s_2 in this thesis. The position vector is defined as:

$$\vec{r}_{J/I}(s_1, s_2) = r_x(s_1, s_2)\hat{\mathbf{i}} + r_y(s_1, s_2)\hat{\mathbf{j}} + r_z(s_1, s_2)\hat{\mathbf{k}} \tag{3.98}$$

where $r_\square(s_1, s_2)$ are the functions used to define the x, y and z displacements of the joint. Because the joint has two DOF, the translational motion space will have two entries:

$$\mathcal{M}_T = \langle \hat{\mathbf{u}}_\mathbf{1}, \hat{\mathbf{u}}_\mathbf{2} \rangle \tag{3.99}$$

where

$$\hat{\mathbf{u}}_\mathbf{1} = \frac{\vec{u}_1}{\|u_1\|} \tag{3.100}$$

and

$$\hat{\mathbf{u}}_{\mathbf{2}} = \frac{\vec{u}_2}{\|u_2\|}. \tag{3.101}$$

u_1 and u_2 can be calculated as

$$\vec{u}_1 = \frac{d\vec{r}_{J/I}(s_1, s_2)}{ds_1} = \frac{dr_x(s_1, s_2)}{ds_1}\hat{\mathbf{i}} + \frac{dr_y(s_1, s_2)}{ds_1}\hat{\mathbf{j}} + \frac{dr_z(s_1, s_2)}{ds_1}\hat{\mathbf{k}} \tag{3.102}$$

and

$$\vec{u}_2 = \frac{d\vec{r}_{J/I}(s_1, s_2)}{ds_2} = \frac{dr_x(s_1, s_2)}{ds_2}\hat{\mathbf{i}} + \frac{dr_y(s_1, s_2)}{ds_2}\hat{\mathbf{j}} + \frac{dr_z(s_1, s_2)}{ds_2}\hat{\mathbf{k}}. \tag{3.103}$$

The corresponding translational reaction space is

$$\mathcal{F}_T = \hat{\mathbf{n}} \tag{3.104}$$

where

$$\hat{\mathbf{n}} = \hat{\mathbf{u}}_{\mathbf{1}} \times \hat{\mathbf{u}}_{\mathbf{2}}. \tag{3.105}$$

The calculation for the joint velocity and acceleration follow the same derivation process shown in Section 3.2.1, except it is extended to include the additional joint coordinate. Using (3.45), the relative velocity between the two constrained bodies can be computed

$$\vec{v}_{J/I} = \dot{\vec{r}}_{J/I}(s_1, s_2) + \vec{\omega}_I \times \vec{r}_{J/I}(s_1, s_2). \tag{3.106}$$

Differentiating again gives the relative acceleration between the two bodies

$$\vec{a}_{J/I} = \ddot{\vec{r}}_{J/I}(s_1, s_2) + \dot{\vec{\omega}}_I \times \vec{r}_{J/I}(s_1, s_2) + \vec{\omega}_I \times \dot{\vec{r}}_{J/I}(s_1, s_2) + \vec{\omega}_I \times \vec{\omega}_I \times \vec{r}_{J/I}(s_1, s_2). \tag{3.107}$$

The Euler angle functions used to define the orientation are defined as

$$\{\theta(s_1, s_2), \phi(s_1, s_2), \psi(s_1, s_2)\} = \{S_\theta(s_1, s_2), S_\phi(s_1, s_2), S_\psi(s_1, s_2)\} \tag{3.108}$$

and they are used to define a rotation matrix in the same way as for the SEK joint, as shown in (3.51). The angular velocity and acceleration enforced across the joint is the same as for the SEK joint, shown in (3.52) and (3.53).

The torque projection presented in the previous section is required in the DEK joint,

but must be extended to compensate for the additional coordinate.

$$\vec{F}_{TP} = \vec{T}_{net} \cdot \vec{p}_1(s_1, s_2)\hat{\mathbf{u}}_\mathbf{1} + \vec{T}_{net} \cdot \vec{p}_2(s_1, s_2)\hat{\mathbf{u}}_\mathbf{2} \tag{3.109}$$

where

$$\vec{p}_1(s_1, s_2) = \frac{\partial\theta(s_1, s_2)}{\partial s_1}\hat{\mathbf{k}} + \frac{\partial\phi(s_1, s_2)}{\partial s_1}\hat{\mathbf{i}}' + \frac{\partial\psi(s_1, s_2)}{\partial s_1}\hat{\mathbf{k}}'' \tag{3.110}$$

and

$$\vec{p}_2(s_1, s_2) = \frac{\partial\theta(s_1, s_2)}{\partial s_2}\hat{\mathbf{k}} + \frac{\partial\phi(s_1, s_2)}{\partial s_2}\hat{\mathbf{i}}' + \frac{\partial\psi(s_1, s_2)}{\partial s_2}\hat{\mathbf{k}}'' \tag{3.111}$$

assuming that a 3-1-3 Euler angle rotation scheme is used.

3.3.2 Surface generation

The path generation for the DEK joint is not as sophisticated as the algorithm used for the SEK joint. Simple fifth-order polynomial surfaces are used to represent the raw numerical data. MATLAB's `poly55` algorithm, which uses the Levenberg-Marquardt least square algorithm [112] is used to generate the surface fit. A polynomial surface is adequate to represent a steered suspension motion space; however it does mean that the DEK joint cannot currently be used to represent more complex surfaces.

3.4 Computer implementation

As mentioned previously, the SEK and DEK joints have been implemented in MapleSim. To allow maximum flexibility, the theory has been implemented directly in the MapleSim source code, rather than the standard user interface. This was accomplished using a script provided by Dr. Chad Schmitke of Maplesoft [99]. The script makes it possible to replace certain components, in this case existing MapleSim joints, with modified source code. After this, it was a simple task to transcribe the theory presented in this chapter into Maple syntax in the source code.

In each joint, nine functions are required. The first three return vectors that describe the translational kinematics: position, velocity and acceleration. The next three are related to the rotational kinematics: the first returns a rotation matrix that describes the body orientation enforced by the joint, and the next two return vectors the define the angular

velocity and acceleration. Two functions are used to define the motion and reaction spaces. The final function computes the force (F_{TP}) required to act along the motion space to handle the torque projection. In the compliant SEK joint, the force function also computes the force generated by the translational bushing models. Furthermore, a tenth function is used to compute the torque generated by the rotational bushing models.

Limitations

Due to the approach of overwriting existing MapleSim constraints, some minor limitations have been introduced that impact the user experience. The path for the SEK joint must be hard-coded into the source file, rather than defined in the user interface as would be expected in a final, production implementation. This has no impact on the mathematics of the joint, but does make it challenging to create a model with multiple SEK joints with different paths.

A limitation in the implementation has been encountered that impacts the mathematical versatility of the joint. If the splines that define each entry in the rotation matrix are very large (i.e. a piecewise polynomial with many pieces), MapleSim can struggle to formulate the model or solve the equations. In any case that this has happened, smaller splines or a single polynomial expression can be substituted into the model and the simulation can proceed with no difficulties, but with slightly reduced accuracy. This demonstrates that there is no fundamental problem with the theory of the SEK joint, only issues related to memory limitations in MapleSim.

Chapter 4

Applications

In this chapter some practical examples of the application of the SEK and DEK joints are shown. The purpose of these examples is to validate the joints in more realistic models than shown in the previous chapters and to demonstrate the flexibility of the SEK joint. Furthermore, it is intended that these examples show the need for such constraints in a number of fields as well as their advantages over more traditional modelling approaches both in terms of model development speed and simulation speed. The first set of examples are from the domain of vehicle dynamics. First, a single MacPherson suspension model is generated and simulated. Next, a full vehicle model is constructed using the MacPherson suspension model. The second example is from the biomechanics domain, and shows a simple simulation of a knee joint. The final example is a roller coaster model, which demonstrates an application of the composite path generation procedure.

4.1 Vehicle models

The main motivation for this work has been to develop reduced suspension models to be used in full vehicle models. As such, suspension models are an obvious first choice for a practical application of the SEK and DEK joints. First, a MacPherson suspension model is generated using data from a model in Adams/Car. The data obtained using Adams/Car replicates the data that would be obtained through a physical Kinematics and Compliance (K&C) test. This makes it quick and easy to go directly from test data to an accurate model, as opposed to the correlation process that is required if a geometric

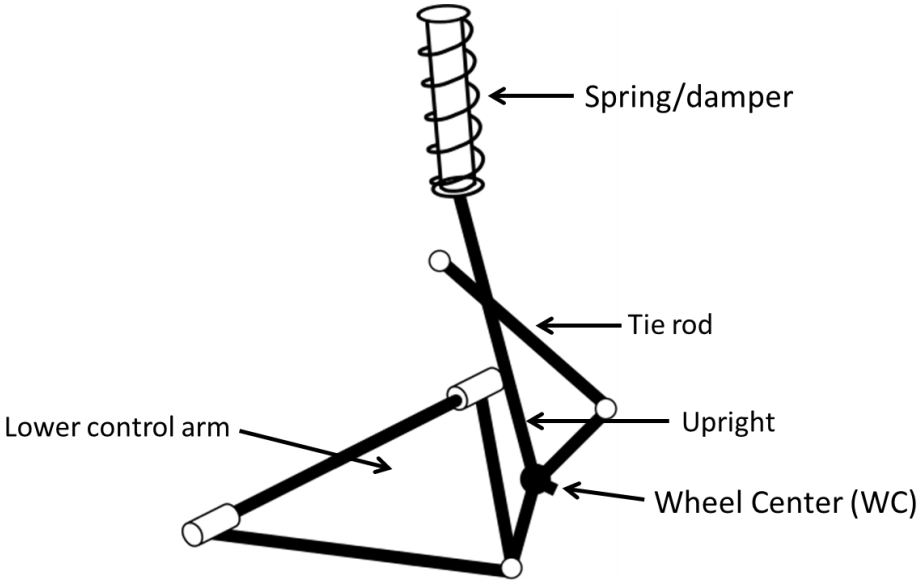


Figure 4.1: MacPherson strut suspension

suspension model is used [53]. First, a rigid version of the MacPherson suspension with no compliance bushings is modeled using the basic SEK joint. Next, bushings are added to the Adams/Car model, and the compliant SEK joint is used to make a model that exhibits similar compliance characteristics to the original high-fidelity model. The first two sets of data are obtained with the steering of the Adams/Car model fixed at 0 degrees. In the third test, additional data is generated at varying steering angles to generate enough data to accurately fit a surface that can be used with the DEK joint. Finally a full vehicle model is constructed using SEK and DEK joints and compared with a high-fidelity model to show both the validity of the new joints and the simulation time improvement.

4.1.1 Rigid suspension

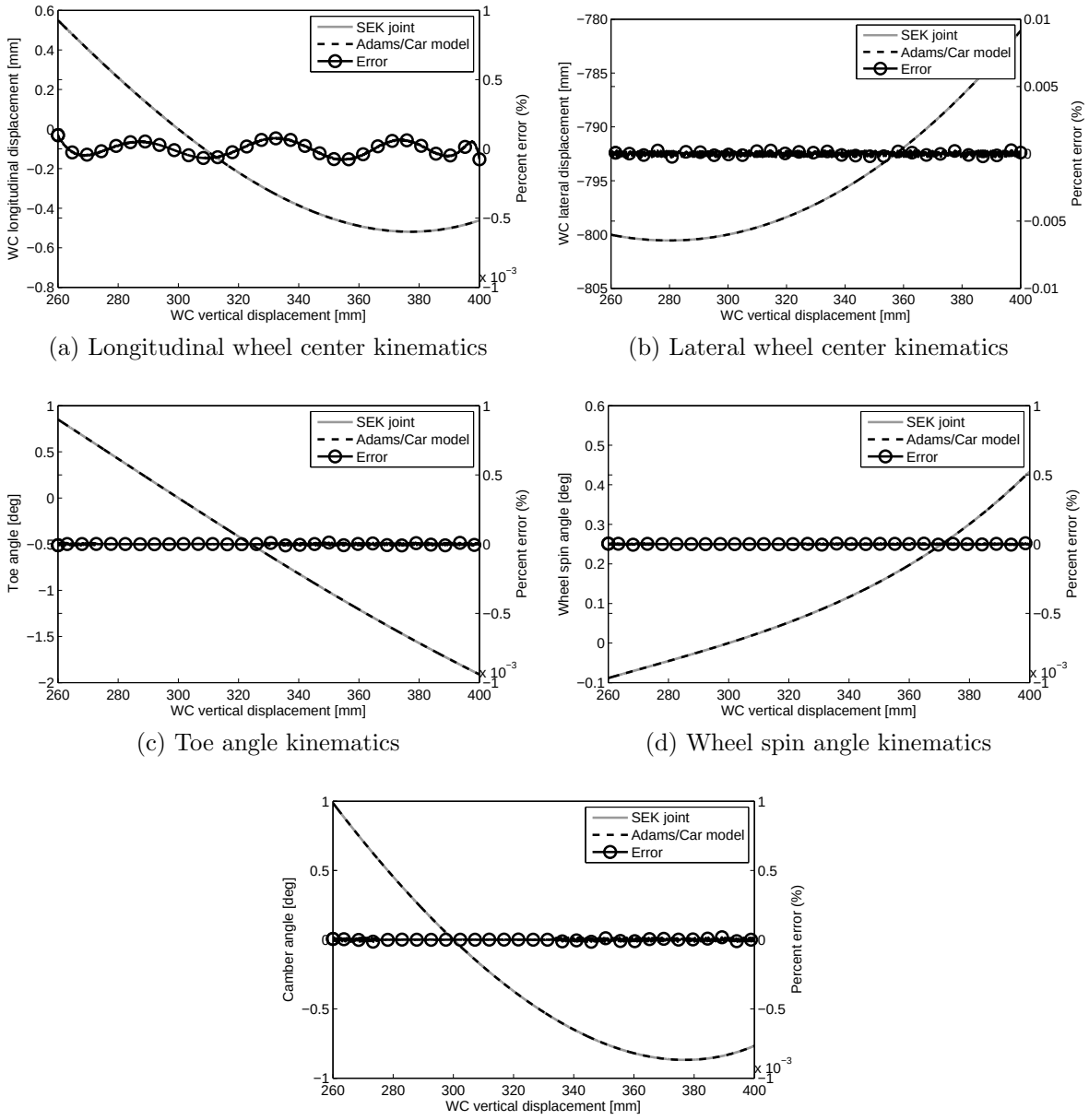
The first model created using the SEK joint is an ideal MacPherson suspension system. A overview of the MacPherson suspension system is shown in Figure 4.1. An equivalent high-fidelity model in the Adams/Car software is used to generate the required kinematic data and provide a target for validation of the SEK suspension model.

The first step in the development of the SEK suspension model is to generate the path required to define the SEK joint. The default MacPherson template from the Adams/Car

2013 "acar_shared" database is used to build the Adams/Car model. To generate the necessary data from the Adams/Car model, requests are created to define the wheel center position and orientation. First, a marker is placed at the wheel center oriented along the wheel spin axis, and on the body of the upright. Next, to compute the wheel center position three requests using the DX(To_Marker, From_Marker, Along_Marker), DY(To_Marker, From_Marker, Along_Marker) and DZ(To_Marker, From_Marker, Along_Marker) functions are created. The To_Marker is defined as the newly created wheel center marker, and the From_Marker and Along_Marker are from the origin of the suspension template. To compute the wheel center orientation is equally straightforward, but care must be taken with regards to the sign of the numbers exported. Using the YAW(To_Marker, From_Marker), PITCH(To_Marker, From_Marker), and ROLL(To_Marker, From_Marker) functions, 3-2-1 Euler Angles for the orientation of the wheel center marker with respect to the suspension template origin can be determined. However, the PITCH() function returns the negated pitch angle [113], so this must be adjusted before the path can be generated. Once the requests are created, a suspension subsystem can be created and used in an assembly, in which the MacPherson subsystem is the only element. The parallel wheel travel simulation is used to generate the data which is then exported from the Adams/Post Processor to be used for the SEK joint path generation.

Once the data is exported from Adams, and correctly formatted, the path generation algorithm can be run. For the initial parameterization step, the vertical displacement, z, is used as the parameterization variable since $z \approx s$. Figure 4.2 shows the results of the path generation procedure. Although the final path is a function dependent on the path length, the results are plotted versus the vertical displacement of the wheel center to facilitate easy comparison with the reference data from Adams/Car.

Once the path has been successfully generated, the next step is to build the MapleSim model that incorporates the SEK joint. The concept of the SEK joint when modeling suspension systems is to remove all of the links and replace them with a single constraint between the chassis and wheel carrier that enforces the correct kinematic and dynamic behaviour. This is a valid assumption since the links have low inertia, and therefore minimal impact on the overall dynamic characteristics of the suspension system. The topology of the model is shown in Figure 4.3. A rigid body is used to represent the unsprung mass (i.e. the upright), and the point on this body that represents the wheel center is constrained to move along the generated path, relative to the chassis, using a



(a) Longitudinal wheel center kinematics

(b) Lateral wheel center kinematics

(c) Toe angle kinematics

(d) Wheel spin angle kinematics

(e) Camber angle kinematics

Figure 4.2: Results of the path fitting algorithm for the rigid MacPherson model. A comparison with the original Adams/Car model is included to show the accuracy of the resulting path.

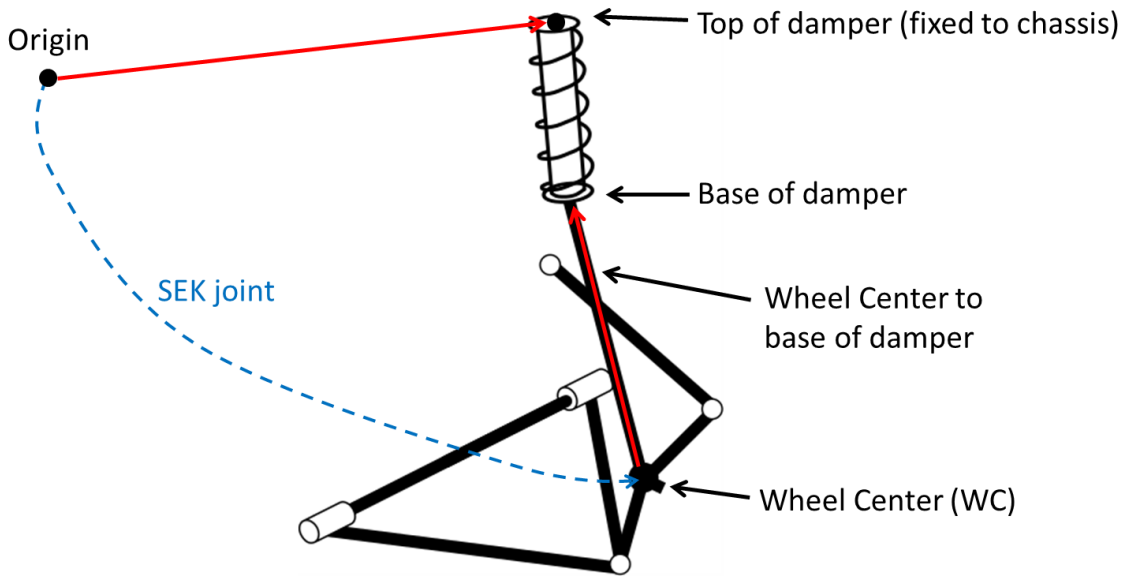


Figure 4.3: Topology of MacPherson suspension model, built in MapleSim using the SEK joint

SEK joint. The SEK joint attaches the unsprung mass to a specific reference point on the chassis; this reference point can be arbitrary, but must be considered when the path is generated. This results in a suspension model with 1-DOF. A spring/damper component is required to complete the model. To ensure an accurate model, the ends of the spring and damper must be connected to the chassis and unsprung mass in the correct locations. The top of the spring/damper assembly is a fixed position on the chassis and can easily be located relative to the origin on the chassis using a rigid body transformation in MapleSim. The base of the spring/damper assembly is a fixed position on the unsprung mass, and likewise can be easily located using a rigid body transformation. The spring/damper assembly is constructed using the Translational Spring Damper Actuator component in MapleSim.

The spring and dampers are modeled as simple linear elements, i.e.

$$F_{spring} = k_{spring}(l - l_0) \tag{4.1}$$

$$F_{damper} = c_{damper} v \tag{4.2}$$

where l is the spring length, l_0 is the unstretched length of the spring, and $v = \dot{l}$ is the

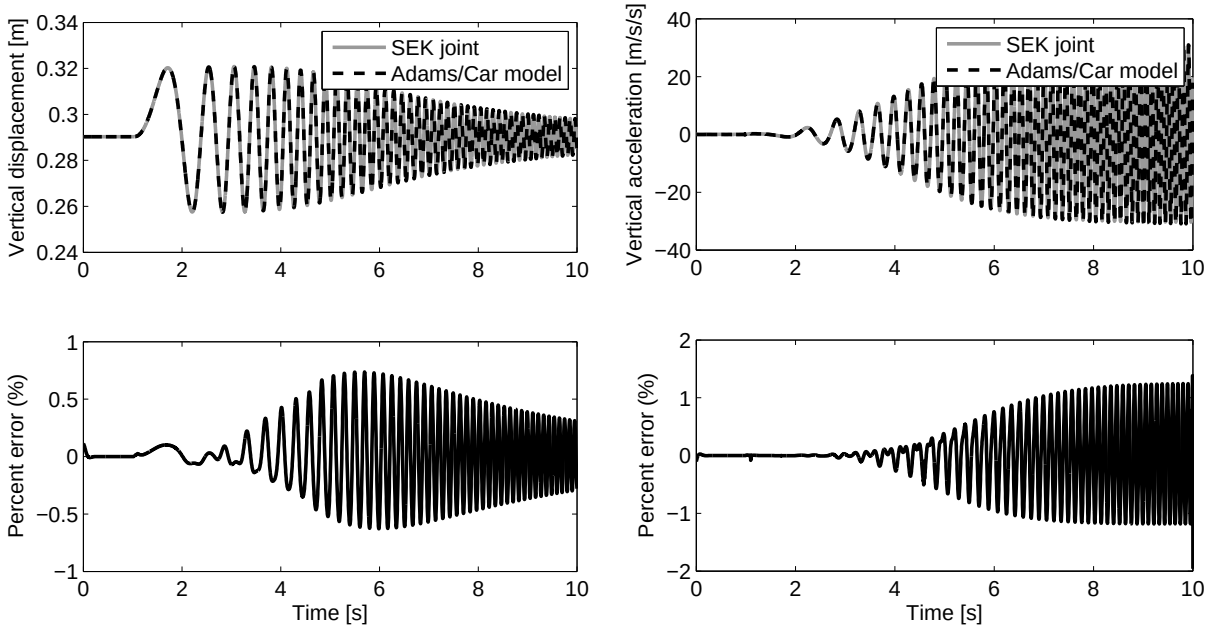
Table 4.1: MacPherson model parameters.

Parameter	Value
Unsprung mass (m_{us})	50 kg
Spring stiffness (k_{spring})	50000 N/m
Spring unstretched length (l_0)	0.3 m
Damping constant (c_{damper})	2200 Ns/m
Unsprung mass inertia (I)	$\begin{bmatrix}1\end{bmatrix}$ kg m^2

damper speed. The model parameters are shown in Table 4.1.

Although the removed links have relatively low mass and inertia, and therefore have minimal impact on the dynamic response of the suspension system, neglecting them in a SEK suspension model means that the two systems are no longer exactly equivalent. As was the case with the spatial four bar examples presented in previous sections, the reference Adams/Car model is modified so that the masses of the links are near-zero to ensure that the behaviour of the SEK suspension system perfectly replicates the Adams/Car model. In practical situations this is not possible, and so a parameter identification scheme should be used to accurately attribute the mass from the links to the unsprung and sprung components of the SEK suspension system. An example of this work with an early version of the SEK joint can be found in [114].

To compare the high-fidelity Adams/Car dynamic model with the reduced SEK joint model, a swept sine vertical force with an amplitude of 1500 N, an offset of 4500 N, and a linear increase in frequency from 0 to 10 Hz that takes 9 seconds and starts at $t = 1s$ is applied to the wheel center. Figure 4.4 shows the resulting position and acceleration of the wheel center during the simulation. It can be seen that the output from the SEK joint model matches the high-fidelity Adams/Car model well. There is some difference between the two result sets (less than 2%) due to differences in the numerical solvers of Adams and MapleSim. These are mainly obvious at high levels of acceleration. A second comparison is made using a step input. The results are shown in Figure 4.5, and as before, the SEK joint suspension model matches the high-fidelity Adams/Car model well.



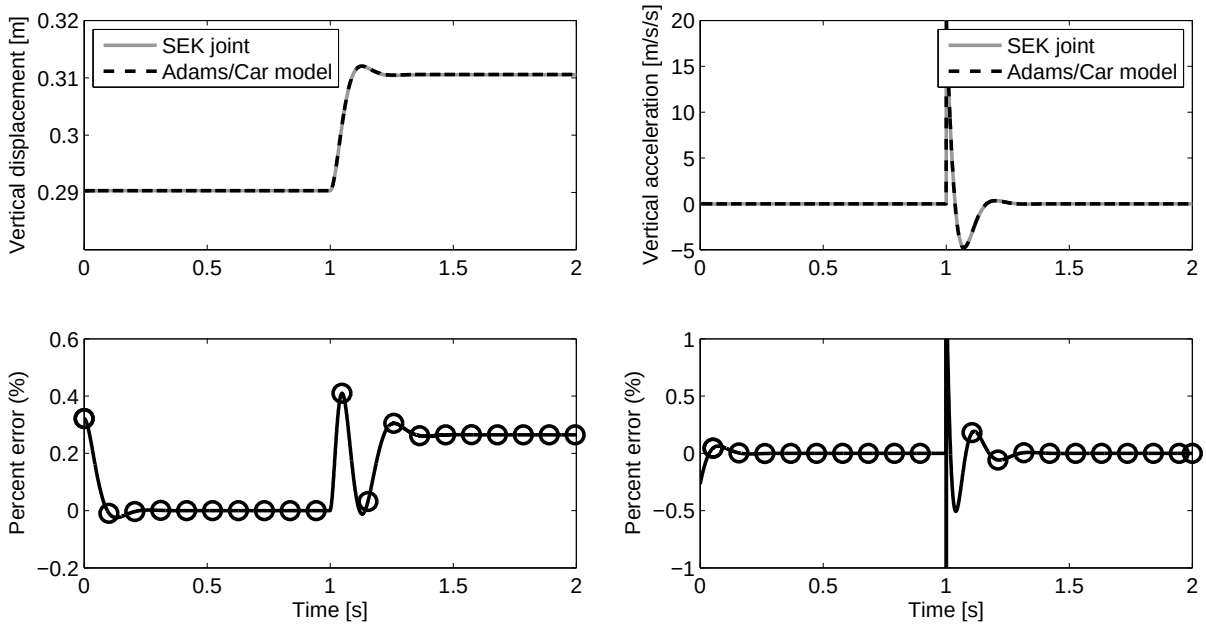
(a) Wheel center vertical displacement versus time (b) Wheel center vertical acceleration versus time

Figure 4.4: Comparison of Adams/Car geometric suspension model and SEK joint characteristic suspension model for a swept sine force input

Simulation speed comparison

A goal for the SEK joint is to offer improved simulation times when compared with conventional high-fidelity geometric suspension models. To ensure an accurate and fair comparison between the two modelling approaches, a third suspension is created: a high-fidelity geometric MacPherson suspension model in MapleSim. Adams/Car does not have a fixed-step solver, and there are no common variable-step solvers between the Adams and MapleSim environments. The high-fidelity MapleSim model is shown to be valid, by comparing it with the SEK joint suspension model, and using the same swept sine vertical force input signal as before. Figure 4.6 shows that the model matches the performance of the SEK joint suspension model. It can be seen that compared to the error shown in Figure 4.4, the error between the high-fidelity geometric MapleSim model and the SEK joint MapleSim model is much lower (less than 0.05% error). This shows that the error in Figure 4.4 is due to differences in the numerical solvers in Adams and MapleSim.

First, to provide a comparison with the Adams software, the default variable-step



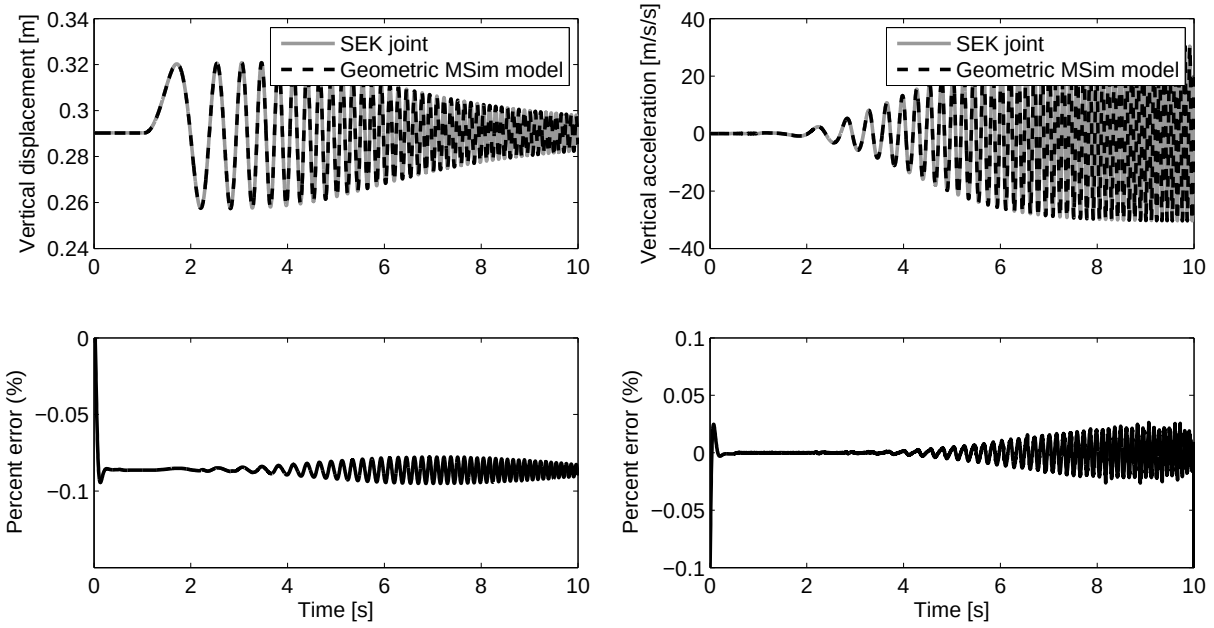
(a) Wheel center vertical displacement versus time (b) Wheel center vertical acceleration versus time

Figure 4.5: Comparison of Adams/Car geometric suspension model and SEK joint characteristic suspension model for a vertical step force input

solver is selected from each package, GSTIFF in Adams and CK45 in MapleSim. To give greater control over the way the simulation is run, the Adams/Car model is exported to Adams/View. This ensures the setup of models are identical. In this case, to compensate for the different solvers and software packages, the point of comparison between the simulation times is the time per function evaluation. Such a measurement provides a consistent means of comparison between each software package and solver; the value is obtained by dividing the simulation CPU time by the number of function evaluations. The results of the comparison between the high-fidelity geometric models in both Adams/View and MapleSim and the MapleSim implemented SEK joint are shown in Table 4.2. The simulation duration is 1000 seconds, and the suspension models are excited with a vertical sinusoidal force input:

$$F_{vertical} = 1500\sin(10\pi t) + 4500 \ \ [\text{N}]. \tag{4.3}$$

The solver error tolerance is set to 10^{-4}. It can be seen that the model constructed using the SEK joint has both the lowest CPU time and time per function evaluation measure.



(a) Wheel center vertical displacement versus time (b) Wheel center vertical acceleration versus time

Figure 4.6: Comparison of MapleSim geometric suspension model and SEK joint characteristic suspension model for a rigid suspension system and a vertical swept sine force input

Since MapleSim is built on the symbolic computing engine of Maple, it is possible to do some investigation to see where the large improvement in simulation time comes from. Looking at the SEK suspension model, there is a single coordinate (s), and a single ODE that is used to describe the system, as expected. The high-fidelity MapleSim suspension model has 6 coordinates that are coupled by 5 algebraic constraints.

A second set of simulations are run comparing only the two MapleSim models and using a fixed step solver. This is desirable because the variation in step size that is present when using a variable step solver can have a large impact on the simulation time. An Euler fixed step solver is used, with a step size of 10^{-4} seconds. The simulation duration is 100 seconds, and the suspension models are excited with the vertical sinusoidal force input from (4.3). The results of this test are shown in Table 4.3. The SEK suspension model implemented in MapleSim runs 6 times faster than the geometric suspension model also implemented in MapleSim.

Table 4.2: Simulation time comparison.

Model	Software	CPU Time [s]	# Fn Evals	Time per Fn. Eval.	Comparison
SEK Joint	MapleSim	1.903	228108	$8.343e-6$	
Geometric	MapleSim	36.161	1816582	$1.991e-5$	2.386x
Geometric	Adams/View	9.519	130882	$7.272e-5$	8.716x

Table 4.3: Simulation time comparison, fixed step solver.

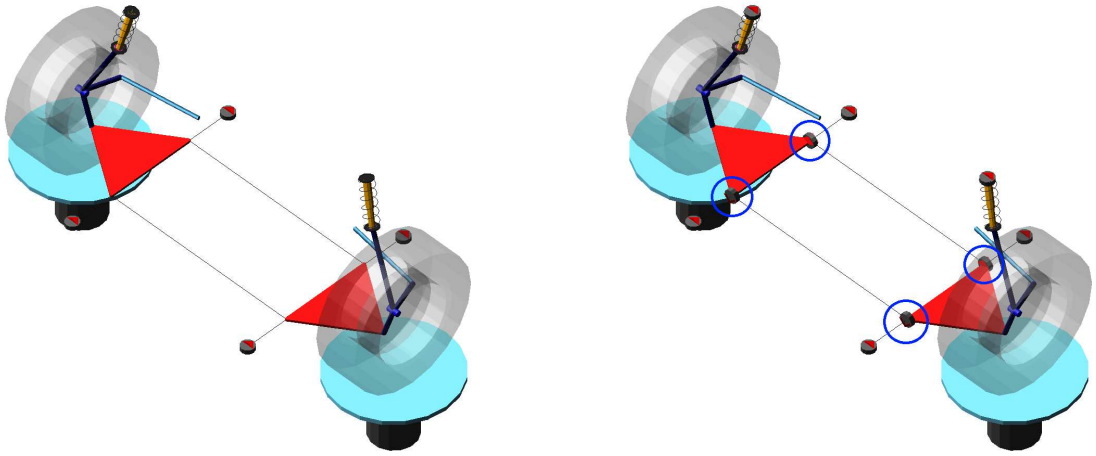
Model	Software	CPU Time [s]	Comparison
SEK Joint	MapleSim	5.132	
Geometric	MapleSim	30.888	6.019x

4.1.2 Compliant suspension

The next example is a compliant version of the same Adams/Car MacPherson model used in the previous section. The Adams/Car model is unchanged, aside from switching the "kinematic-toggle" mode from "kinematic" to "compliant". Now the revolute joint that attaches the lower control arm to the chassis has been replaced with two bushings as shown in Figure 4.7.

The MapleSim model is now constructed using the compliant SEK joint, but the topology is the same as shown in Figure 4.3. A new path is generated using data from the compliant Adams/Car MacPherson model as the addition of the bushings changes the kinematics slightly.

The ten bushing stiffness and damping coefficients must be determined. There are two translational spring stiffnesses, and two damping coefficients, and three rotational spring stiffnesses and three rotational damping coefficients. Because the model topology between the geometric and characteristic models are completely different, the bushing properties from the Adams/Car model cannot be used directly in the compliant SEK joint. Fortunately, in this case the model is relatively simple and it is therefore possible to identify the correct parameter values manually. To identify the compliance characteristics, five simulations are run. The first two simulations are used to identify the translational



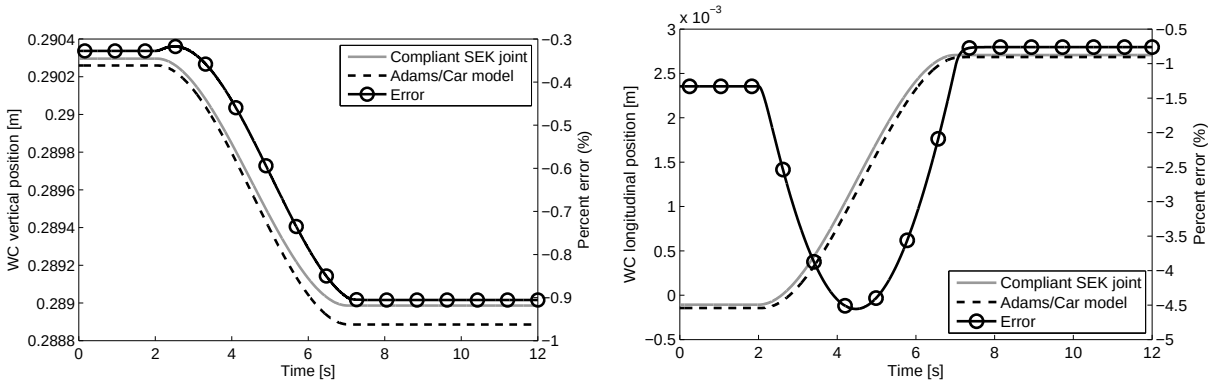
(a) Adams/Car model in kinematic mode with no bushings

(b) Adams/Car model in compliant mode with bushings on the lower control arm

Figure 4.7: Comparison of Adams/Car MacPherson suspension model in kinematic and compliant mode

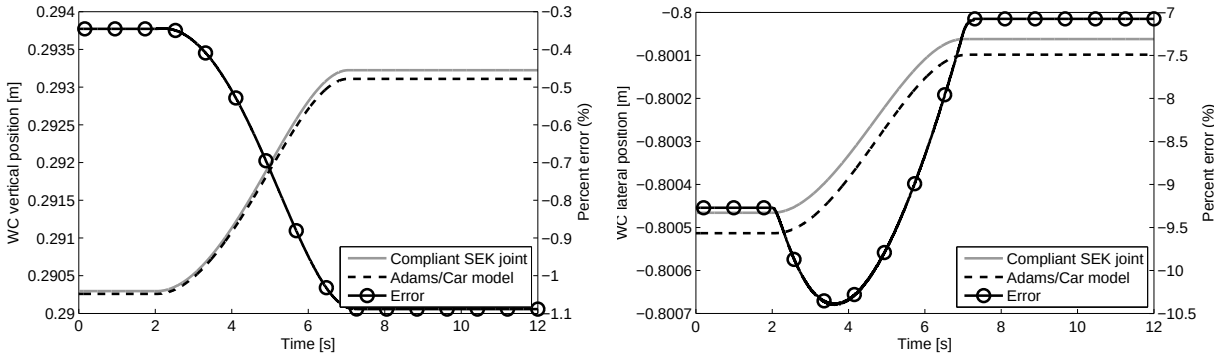
bushing parameters in the normal and binormal directions. This is done by applying a ramp force along the longitudinal x-axis, in the first simulation, a ramp force along the lateral y-axis in the second. The ramp signal is created using the Adams STEP command and goes from 0 N to 4000 N over a 5 second period, starting at $t = 2$ s. The results of these first two tests, using the final identified parameters are shown in Figures 4.8 and 4.9. It can be seen that the results of the compliant SEK joint match those of the Adams/Car model well. This type of input (function and direction of application) is the same as used during K&C tests to measure suspension compliance [77].

The second set of simulations that are run are used to identify the rotational compliance parameters. Three simulations are run: in the first, a ramp moment is applied about the camber (x) axis; in the second a ramp moment is applied about the wheel spin (y) axis; and in the third a ramp moment is applied about the toe (z) axis. The ramp signal is created using the Adams STEP command and goes from 0 Nm to 4000 Nm over a 5 second period, starting at $t = 2$ s. The results are shown in Figures 4.10 to 4.12, and the compliant SEK joint matches the Adams/Car MacPherson model well. There are some small discrepancies between the two models. This is caused by the fact that the compliance in the SEK joint is a simple linear model, whereas the compliance in the full suspension model (when measured at the wheel center) is a non-linear function of the suspension geometry. The identified



(a) Vertical position of wheel center (b) Longitudinal position of wheel center

Figure 4.8: Longitudinal compliance of MacPherson suspension model when subjected to a ramp longitudinal force, after parameter identification

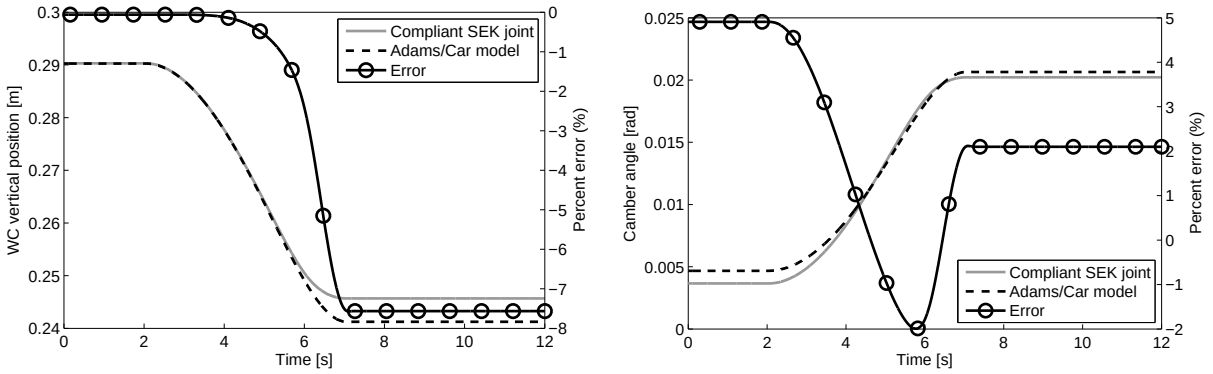


(a) Vertical position of wheel center (b) Lateral position of wheel center

Figure 4.9: Lateral compliance of MacPherson suspension model when subjected to a ramp lateral force, after parameter identification

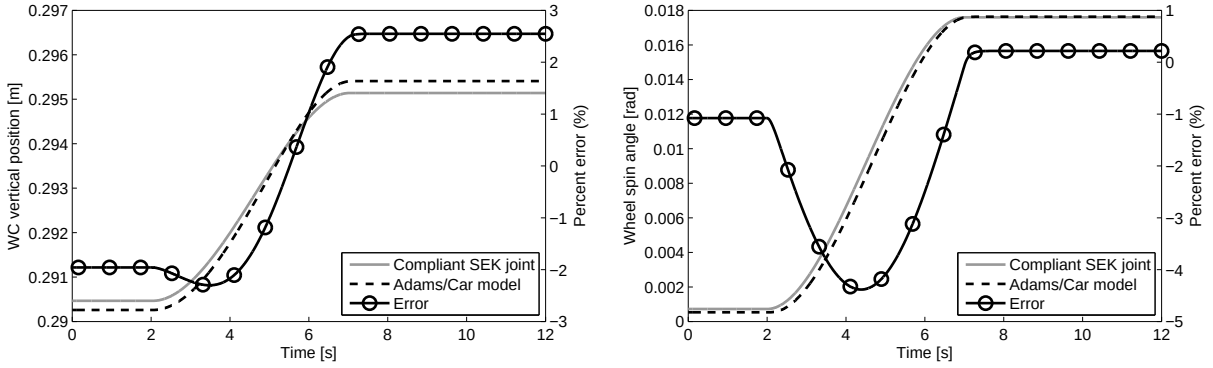
parameters are shown in Table 4.4. The damping values were not identified and were set to 1% of the stiffness value as suggested in [115].

To show that the unconstrained dynamics of the compliant SEK joint suspension model match those of the Adams/Car model, a similar test to the one used in the previous section is carried out. The same swept sine vertical force used in Section 4.1.1 is applied to the wheel center. The results are shown in Figure 4.13, and as before the compliant SEK joint displays the same response to the input as the Adams/Car model.



(a) Vertical position of wheel center (b) Camber angle

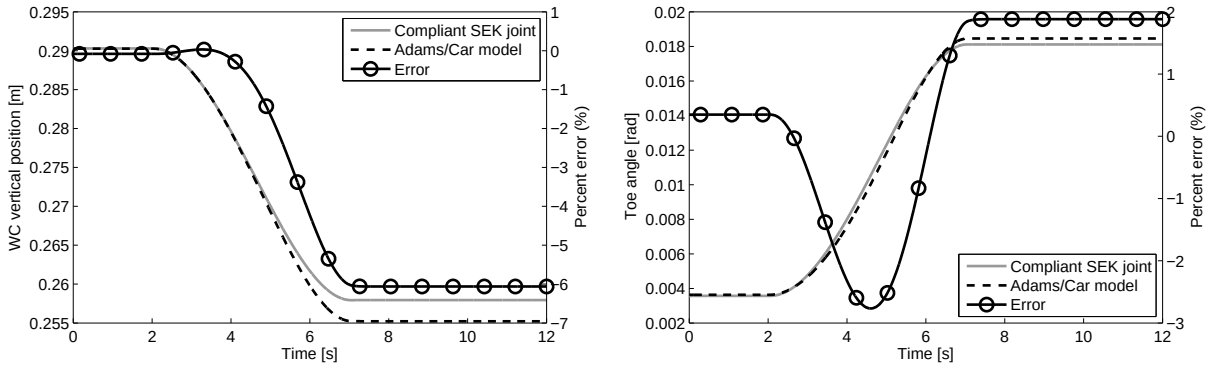
Figure 4.10: Camber compliance of MacPherson suspension model when subjected to a ramp moment about the longitudinal axis, after parameter identification



(a) Vertical position of wheel center (b) Wheel spin angle

Figure 4.11: Wheel spin compliance of MacPherson suspension model when subjected to a ramp moment about the lateral axis, after parameter identification

As with the rigid suspension models, the major goal of the compliant SEK joint suspension model is to offer faster simulation times when compared with more conventional geometric models. First a comparison is made using the variable time step solvers in both Adams and MapleSim. The default variable-step solver is selected from each package, GSTIFF in Adams and CK45 in MapleSim. As in the previous section the point of comparison between the simulation times is the time per function evaluation to give the most fair comparison despite the different solvers. The simulation duration is 1000 seconds, and the suspension models are excited with the vertical sinusoidal force input from (4.3). The



(a) Vertical position of wheel center

(b) Toe angle

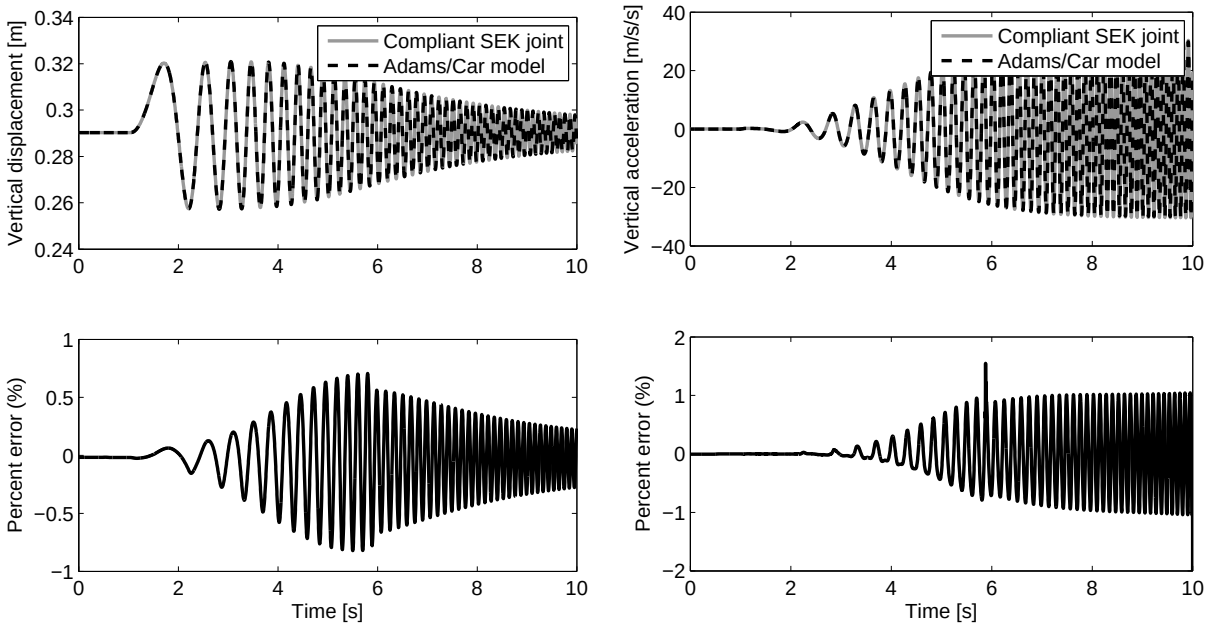
Figure 4.12: Toe compliance of MacPherson suspension model when subjected to a ramp moment about the vertical axis, after parameter identification

Table 4.4: Identified compliance parameters for compliant SEK MacPherson suspension model

Direction	Compliance stiffness	Compliance damping
$\hat{n}$	1300 kN/m	13 kN s / m
$\hat{b}$	1.45 kN/m	0.0145 kN s / m
$\hat{\theta}$	187.5 kN/rad	1.875 kN s / rad
$\hat{\phi}$	227.5 kN/rad	2.275 kN s / rad
$\hat{\psi}$	2500 kN/rad	25 kN s / rad

solver error tolerance is set to 10^{-4}. The results of the comparison between the compliant high-fidelity geometric models in both Adams/View and MapleSim and the MapleSim implemented compliant SEK joint are shown in Table 4.5. As with the rigid suspension models it can be seen that the model constructed using the SEK joint has the lowest time per function evaluation measure. Once again, it is possible to compare the number of equations used in the geometric MapleSim model and the compliant SEK MapleSim model. Looking at the SEK suspension model, there are six coordinates (s, n, b, θ_d, ϕ_d and ψ_d), and 6 ODEs are used to describe the system, as expected. The high-fidelity suspension model has 15 coordinates that are coupled by 8 algebraic constraints.

A second set of simulations are run comparing only the two MapleSim models and using



(a) Wheel center vertical displacement versus time (b) Wheel center vertical acceleration versus time

Figure 4.13: Comparison of Adams/Car geometric model and SEK joint characteristic model for a compliant suspension system and a vertical swept sine force input

a fixed step solver. An Euler fixed step solver is used, with a step size of 10^{-4} seconds. The simulation duration is 100 seconds, and the suspension models are excited with the vertical sinusoidal force input from (4.3). The results of this test are shown in Table 4.6. The SEK suspension model implemented in MapleSim runs almost 3 times faster than the high-fidelity geometric suspension model also implemented in MapleSim.

Table 4.5: Simulation time comparison.

Model	Software	CPU Time [s]	# Fn Evals	Time per Fn. Eval.	Comparison
SEK Joint	MapleSim	38.018	3426121	$1.110e-5$	
Geometric	MapleSim	136.017	7021903	$1.937e-5$	1.745x
Geometric	Adams/View	23.06	254613	$9.057e-5$	8.159x

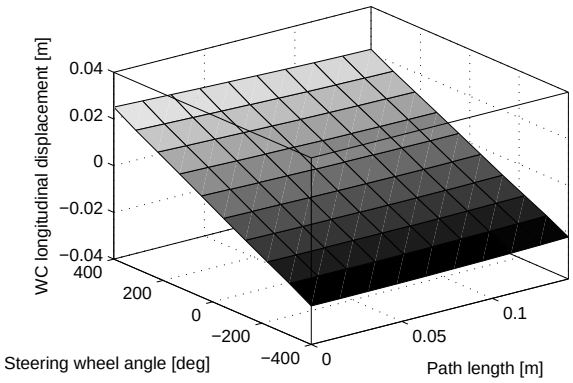
Table 4.6: Simulation time comparison, fixed step solver.

Model	Software	CPU Time [s]	Comparison
SEK Joint	MapleSim	26.536	
Geometric	MapleSim	72.322	2.725x

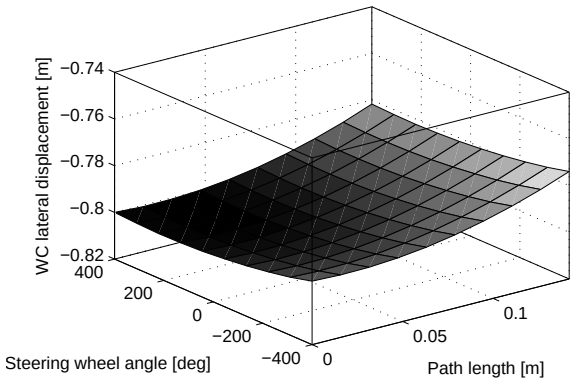
4.1.3 Rigid steered suspension

The third suspension example uses the DEK joint to represent a steered version of the same rigid MacPherson suspension model that was used in the first example. A steered suspension has two degrees of freedom; the first is the vertical travel, as in an unsteered suspension, and the second is the steering actuation. Consequently, rather than following a spatial curve, the possible motion space for the wheel center follows a surface. As before, the data for the surface is generated using the Adams/Car MacPherson model. This time, multiple parallel wheel travel simulations are carried out with different fixed steering positions. The steering positions used are defined using the steering wheel angle and range from $-400°$ to $400°$ with increments of $100°$. As before, the wheel center longitudinal and lateral displacements are measured, as are the toe, camber and wheel spin angles. The results of the surface fitting procedure are shown in Figure 4.14.

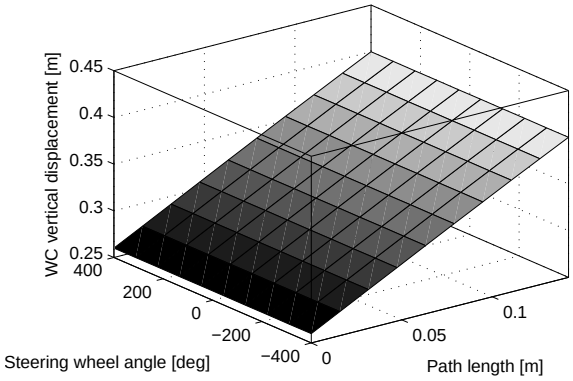
Once the surface is fit, the next step is to build the MapleSim model of the MacPherson suspension using the DEK joint. The exact model topology used for the SEK joint examples cannot be used as the joint now has an additional degree of freedom that must be constrained—the steering angle. There are two different options for how to build the model. The first is a hybrid of the reduced characteristic and high-fidelity geometric approaches discussed in Chapter 1. In this multibody model, the tie rod is included in exactly



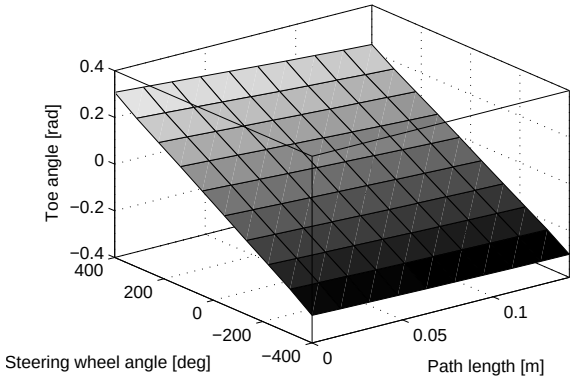
(a) Longitudinal kinematics



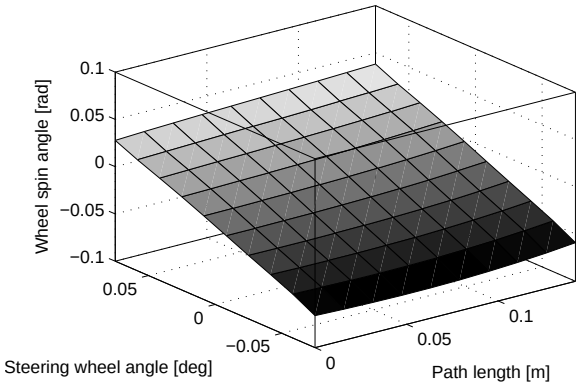
(b) Lateral kinematics



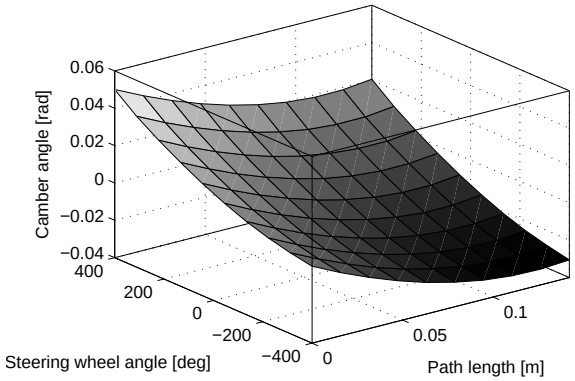
(c) Vertical kinematics



(d) Toe angle kinematics



(e) Wheel spin angle kinematics



(f) Camber angle kinematics

Figure 4.14: Results of the surface fitting algorithm for the rigid steered MacPherson model

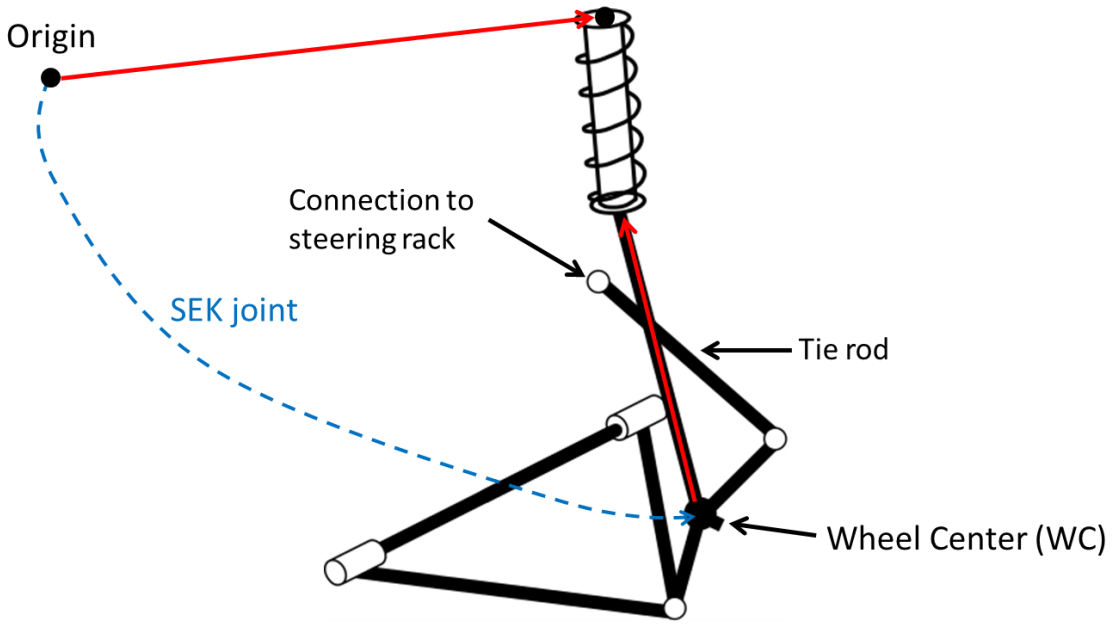


Figure 4.15: Topology of steered MacPherson suspension model, built in MapleSim using the DEK joint

the same way that it would be in a geometric suspension model. This is illustrated in Figure 4.15. The tie rod is modelled as a rigid body that is connected to the chassis using a spherical joint, and connected to the unsprung mass using a universal joint. Each end of the tie rod is located using a rigid body transformation to describe vectors from the chassis origin and the wheel center. Using the Chebyshev-Grübler-Kutzbach criterion [116] it can be seen that the system now has a single DOF:

$$M = 6(2) - 4 - 4 - 3 = 1 \tag{4.4}$$

since there are two rigid bodies (unsprung mass and tie rod), the DEK joint and universal joint constrain 4 DOF and the spherical joint constrains 3 DOF. If the steering is to be actuated, then further complexity must be added to include a 1 DOF translational component attached to the spherical joint on the tie rod to represent the steering rack. This approach, while certainly valid, does violate some of the goals for the DEK joint approach: specifically, there is now a closed kinematic chain and the system is now described using a set of DAEs rather than pure ODEs.

An alternative approach is to use a topology similar to that used for the SEK joint

Table 4.7: Simulation time comparison DEK suspension model alternatives, fixed step solver.

Model	Software	CPU Time [s]	Comparison
DEK Joint	MapleSim	6.755	
Geometric	MapleSim	30.888	4.572x

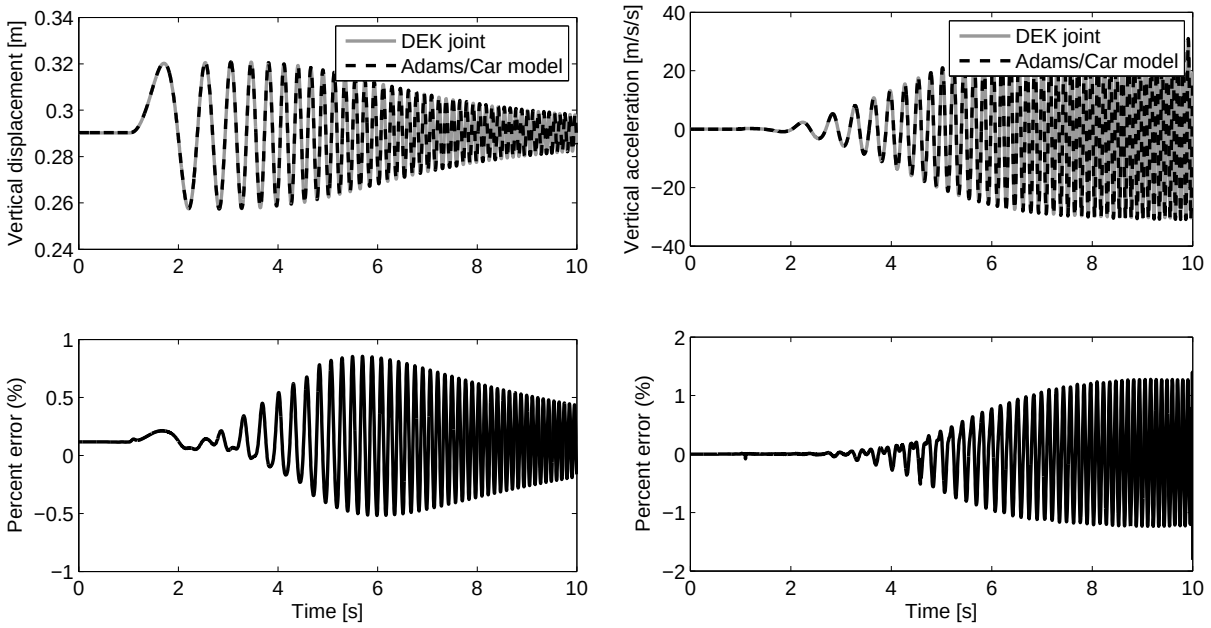
models (Figure 4.3) and place a driving constraint on the steering angle coordinate. This is more in line with the "characteristic" approach to modelling the suspension system, and results in a much simpler model with no closed kinematic chains. The disadvantage to this approach is that the model does not include a multibody representation of the steering system, meaning it could not be used for certain studies where the dynamics of the steering system are of special interest.

Having discussed the topology differences between the two modelling approaches, the next obvious point of comparison is the simulation times. Table 4.7 shows the simulation times for a 100 second simulation using the sinusoidal vertical force input from (4.3) with fixed straight-ahead steering. The solver used is the MapleSim Euler fixed-step solver with a step size of $1e-4$ seconds. It can be seen that the second approach of using a numerical constraint to control the steering degree of freedom gives much faster simulation times than using a multibody approach. Also included in the table is the result of running the simulation using a high-fidelity geometric model. The multibody approach of constraining the DEK eliminates all of the speed advantages of the characteristic modelling approach.

As before, it is possible to compare the number of generated equations for the two models using the DEK joint. The model with the multibody tie rod representation has 4 coordinates that are coupled by 3 algebraic constraints. The model with the numerical constraint is represented using two coordinates (s_1 and s_2) coming from the DEK joint. Two ODEs are used, and there is an algebraic constraint on one of the coordinates, s_2 in this case—the steering angle.

It was therefore decided to proceed with the second approach to the steering for the rest of this example. First, a simple validation is performed using the same swept sine input as in the previous examples. In this case the steering is held fixed at 0°. This validates the dynamics of the DEK suspension model against the Adams/Car suspension

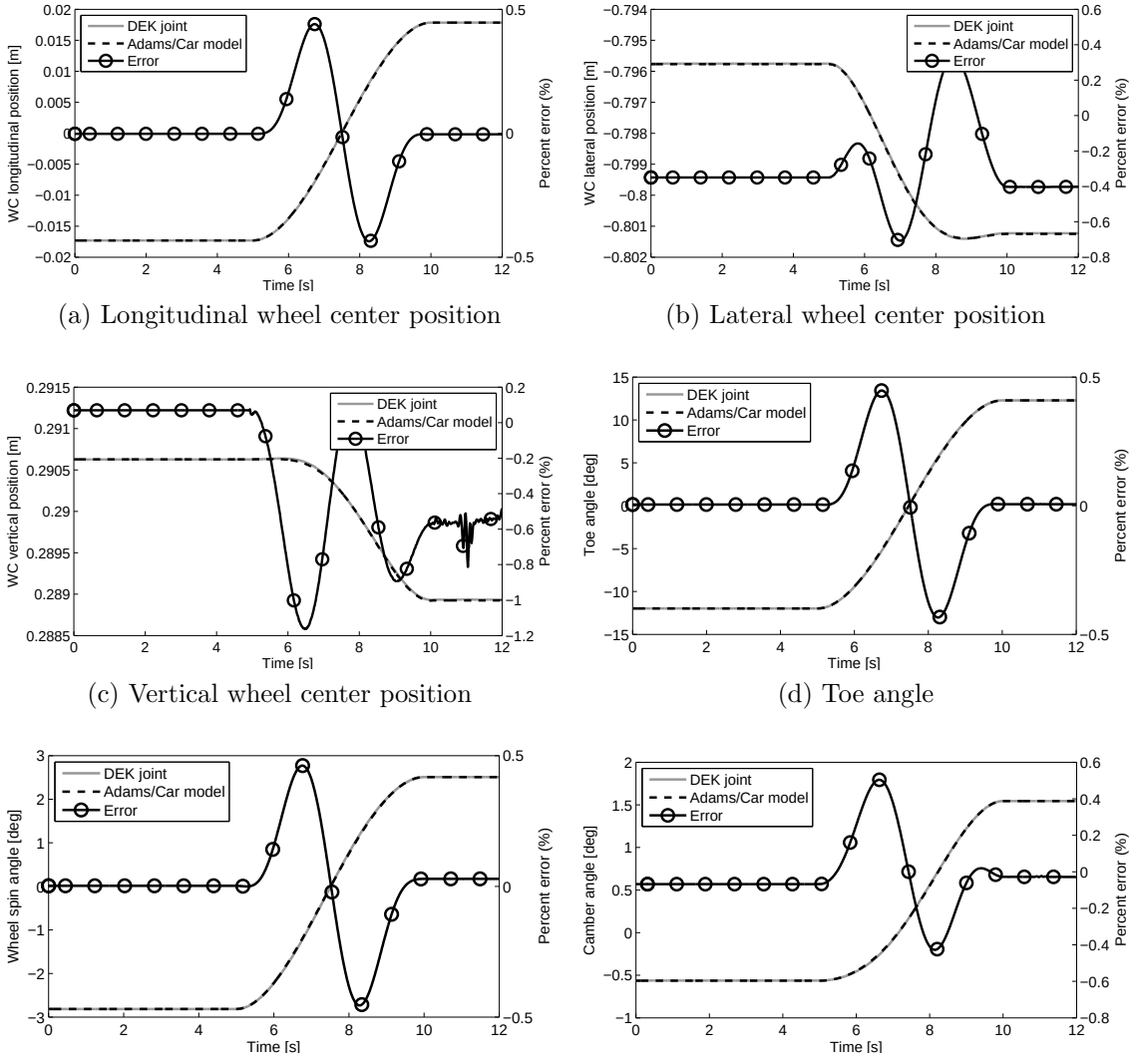
model from which the kinematic data was generated. The results of this comparison are shown in Figure 4.16. The behaviour of the DEK joint suspension model matches that of the Adams/Car suspension model closely.



(a) Wheel center vertical displacement versus time (b) Wheel center vertical acceleration versus time

Figure 4.16: Comparison of Adams/Car geometric model and DEK joint characteristic model for a rigid suspension system and a vertical swept sine force input

The second test is to show the accuracy of the steering kinematics. A constant vertical force of 4500 N is applied to the wheel center, while a ramp input is used to move the steering wheel angle from $-300°$ to $300°$. The wheel center kinematics resulting from this input are shown in Figure 4.17. The results show that the response of the DEK joint suspension model matches the response of the Adams/Car model to the same input.



(a) Longitudinal wheel center position

(b) Lateral wheel center position

(c) Vertical wheel center position

(d) Toe angle

(e) Wheel spin angle

(f) Camber angle

Figure 4.17: Comparison of Adams/Car geometric model and DEK joint characteristic model for a rigid suspension system and a ramp steering displacement input

4.1.4 Full vehicle model

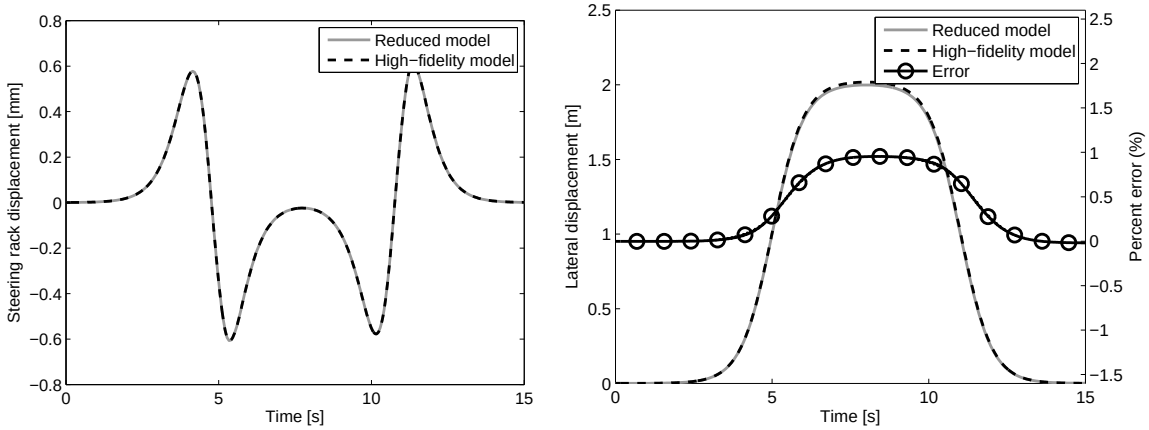
The three previous examples showed subsystem level applications of the SEK and DEK joints. In this example, multiple SEK and DEK joints are used to construct a full vehicle model with four wheels. The front suspension is constructed using the DEK representation of the steered MacPherson suspension system. The rear suspension is constructed using the rigid SEK representation of the MacPherson suspension system. The chassis is represented using a single rigid body. An equivalent geometric high-fidelity model is created in MapleSim to provide a comparison point in terms of numerical accuracy and simulation speed. Unlike the previous examples, the suspension links in the high-fidelity model have mass. In the reduced model, since the links are neglected, their mass must be distributed between the sprung and unsprung masses. For this example, the mass is simply distributed evenly; however more advanced techniques can be used to determine the optimal redistribution of the masses [114]. Both models use a Fiala tyre model. The DOF for each model is sixteen. Six DOF come from the chassis, four represent the vertical motion of each suspension, four represent the wheel spins and the final two are for the steering on each front suspension. The steering is prescribed so that the car follows a 2 meter double lane change maneuver as shown in Figure 4.18. The suspension hard points for the high-fidelity geometric models can be found in [117]. The model parameters are shown in Table 4.8.

The results of a comparative double lane change simulation are now shown. In both cases a fixed step Euler solver was used, with a step size of $1e-4$ seconds. The duration of the simulation is 15 seconds. Figure 4.18a shows the steering wheel input that was used for each model. This input was obtained using the high-fidelity model and a simple PID path following driver model. Once the steering input trace was obtained, it was used in a lookup table to control each model so that a fair comparison could be made between the two models. In both cases, a simple PID controller is used to apply torque at the rear wheels to maintain a speed of 80 km/h throughout the maneuver. Figure 4.18b shows the lateral displacement of each model when subjected to the steering input. It can be seen that the reduced model tracks the results obtained from the high-fidelity model closely, but not perfectly, which is expected given the minor differences between the two models. Figure 4.19 shows that the roll and yaw velocities of the chassis are similar for each model, and Figure 4.20 shows that the normal forces for each of the four tyres matches almost perfectly (less than 0.2% error) between the two models.

Table 4.8: Full vehicle model parameters.

Parameter	Value	
Chassis mass	1780	kg
Wheel carrier mass	20	kg
Lower control arm mass	5	kg
Tie rod mass	5	kg
Wheel base	2.58	m
Front axle to chassis CG	1.23	m
Chassis CG height	0.45	m
Chassis inertia	$\begin{bmatrix} 200 & 0 & 0 \\ 0 & 500 & 0 \\ 0 & 0 & 600 \end{bmatrix}$	kg m^2
Wheel/tyre mass	25	kg
Wheel/tyre inertia	$\begin{bmatrix} 0.8 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	kg m^2
Tyre spring stiffness	310000	N/m
Tyre damping constant	3100	Ns/m
Tyre unloaded radius	0.31	m
C_α	800	N/deg
C_{slip}	1000	N
Spring stiffness (k_{spring})	50000	N/m
Spring unstretched length (l_0)	0.3	m
Damping constant (C_{damper})	2200	Ns/m
Unsprung mass inertia ($I_{unsprung}$)	$\begin{bmatrix} 1 \end{bmatrix}$	kg m^2

As before, the second point of comparison between the reduced model and the high-fidelity model is the number of resulting equations and the subsequent simulation time. The high fidelity model is modelled with 34 coordinates that are coupled by 18 algebraic constraints. On the other hand, the model constructed using the SEK and DEK joints uses 16 coordinates, and 16 ODEs, with 2 driving functions from the DEK steering constraint. As previously mentioned, the same Euler fixed-step solver was used for both models with a step size of $1e-4$ seconds. A comparison of the simulation times is shown in Table 4.9. The simulation of the reduced model with the SEK suspension joints runs 2.4 times faster than the simulation using the high-fidelity geometric model.

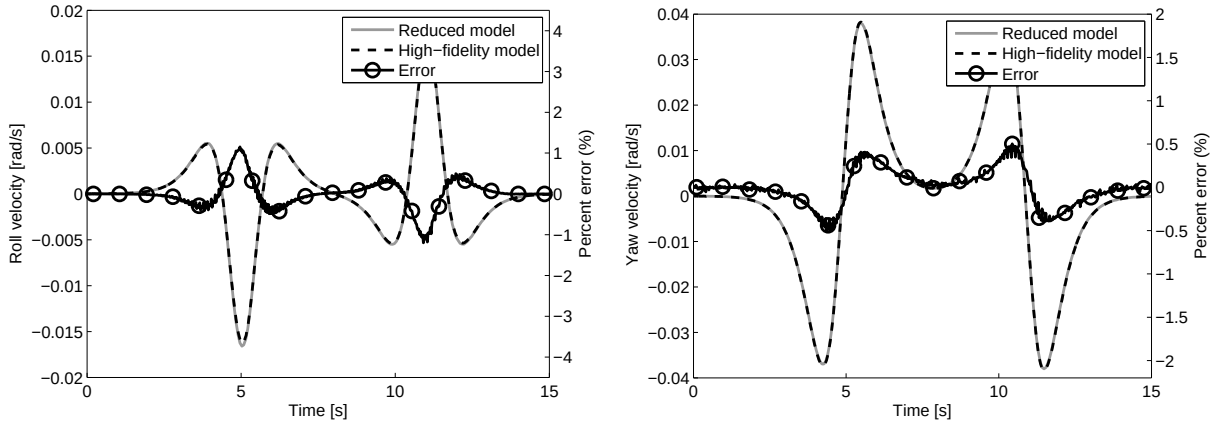


(a) Steering rack displacement versus time (b) Vehicle lateral displacement versus time

Figure 4.18: Steering rack displacement and resulting lateral displacement comparison for high-fidelity and reduced vehicle models during a double lane change maneuver

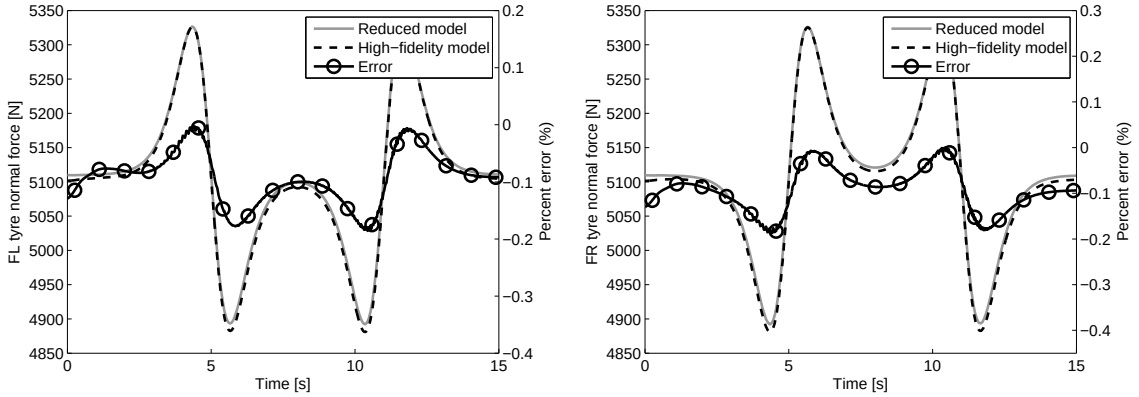
Table 4.9: Simulation time comparison full vehicle models, fixed step solver.

Model	Software	CPU Time [s]	Comparison
SEK/DEK Joints	MapleSim	4.898	
High fidelity geometric	MapleSim	11.778	2.405x



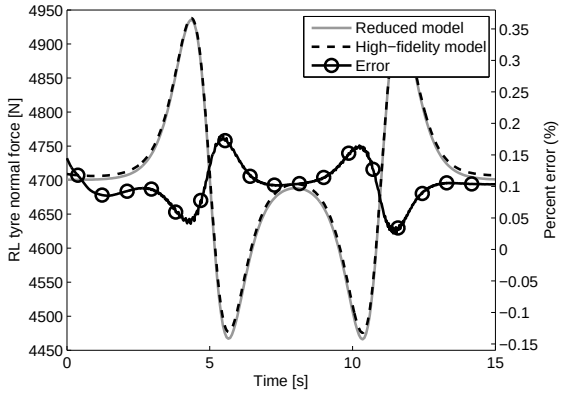
(a) Roll velocity versus time (b) Yaw velocity versus time

Figure 4.19: Roll and yaw velocity comparison for high-fidelity and reduced vehicle models during a double lane change maneuver

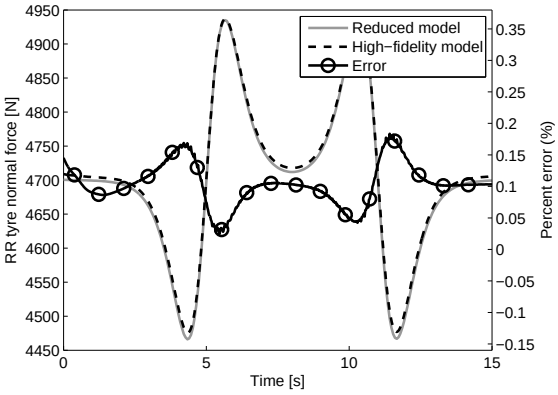


(a) Front left tyre normal force versus time

(b) Front right tyre normal force versus time



(c) Rear left tyre normal force versus time



(d) Rear right tyre normal force versus time

Figure 4.20: Tyre normal forces comparison for high-fidelity and reduced vehicle models during a double lane change maneuver

4.2 Knee

The next example comes from the field of biomechanics. A knee model is created using the SEK joint. The data is obtained from the work done by DeFrate et al. [118] in which they used fluoroscopy to measure the kinematics of the knee joint during a lunge maneuver and compare people with ACL injuries and healthy knees. In this example, the kinematics of the healthy knees were used to construct the path for the SEK joint. Five sets of data were provided in the work, all as a function of the flexion angle of the knee. There are three translations: the anterior-posterior (AP) translation, the medial-lateral (ML) translation and the superior-inferior (SI) translation, and two rotations: the internal-external (IE) rotation and the valgus-vargus (VV) rotation. The coordinate system is shown in Figure 4.21. From this data it is possible to construct the path functions needed to define the SEK joint. Generally, a knee is modelled with the flexion angle as the independent variable, but in the SEK joint, the independent variable is the path length. This is possible because the tibia translates as well as rotates relative to the femur as the knee is bent, meaning it is following a spatial path. Figure 4.22 shows the results of the path generation algorithm using the data from [118].

Compared to other modelling techniques for the human knee, the SEK joint approach does offer some advantages, but also disadvantages and so is not the best choice in all cases. Compared with the simplest and most common approach of using a revolute joint to model the knee joint as an ideal revolute joint, the SEK joint offers more accurate kinematics, without a penalty in simulation time. Other approaches such as contact models and finite element approaches require significantly more information to build a model and are much slower to simulate, but result in a model that can be used in a variety of situations and provide more useful information on the contact forces and moments within the knee joint. For example, the data used in this example is from a lunge; however during another maneuver (such as walking) the forces acting on the knee joint will be different, resulting in different knee kinematics, and meaning that a different path would be required for the SEK joint. Using the compliant SEK joint could go someway to removing this limitation; however it is challenging to obtain detailed and accurate compliant data for human knees.

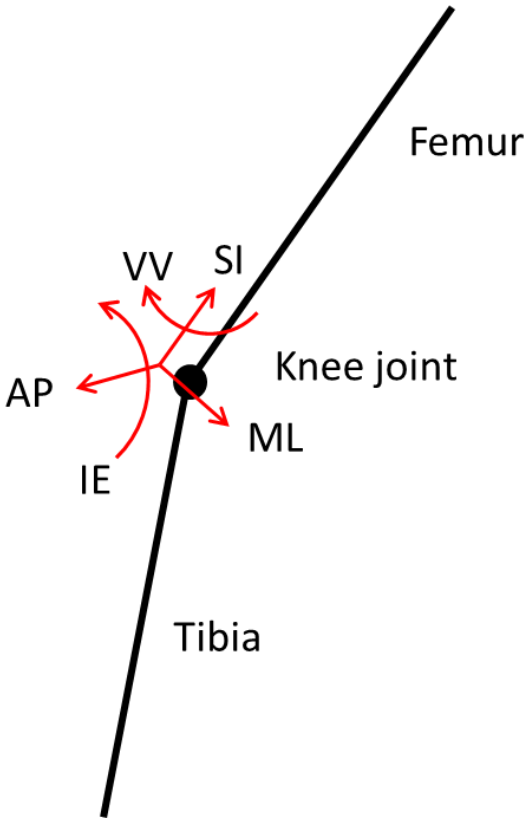
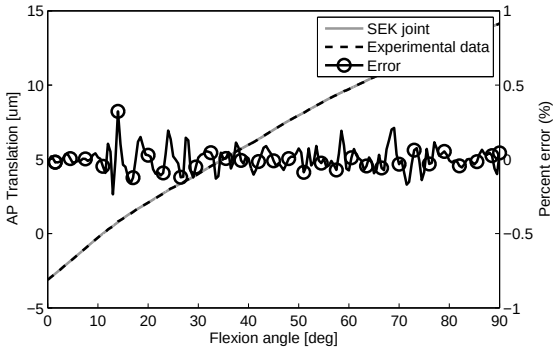
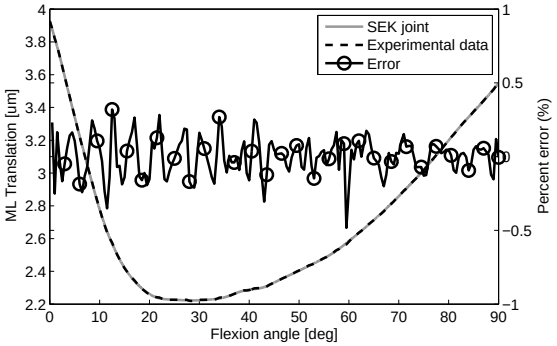


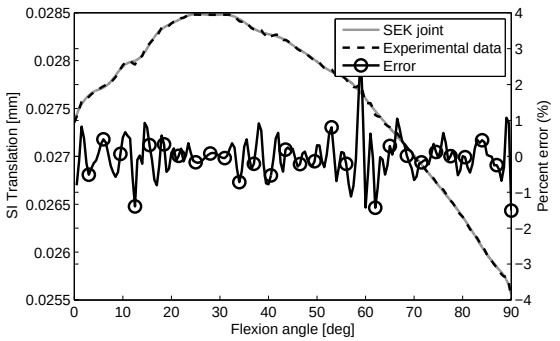
Figure 4.21: Coordinate system fixed on the femur used to model the knee kinematics



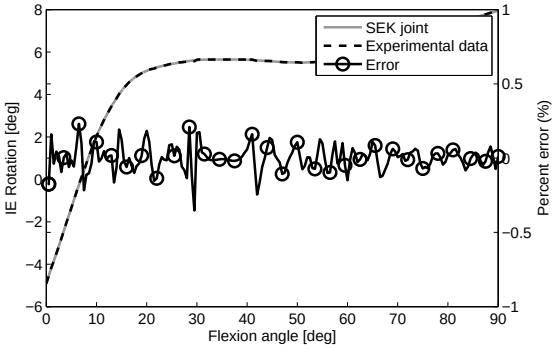
(a) Tibia anterior-posterior translation relative to the femur as a function of knee flexion angle



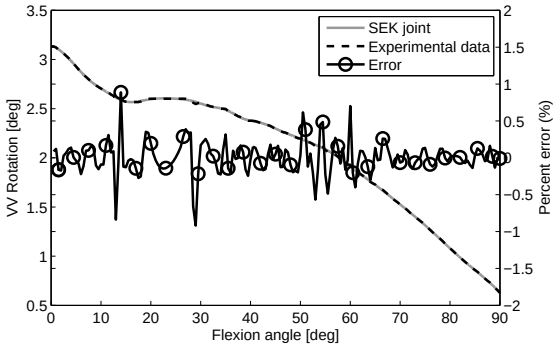
(b) Tibia medial-lateral translation relative to the femur as a function of knee flexion angle



(c) Tibia superior-inferior translation relative to the femur as a function of knee flexion angle



(d) Tibia internal-external rotation relative to the femur as a function of knee flexion angle



(e) Tibia valgus-vargus rotation relative to the femur as a function of knee flexion angle

Figure 4.22: Comparison of SEK knee joint kinematics with experimental data

4.3 Roller coaster

The final example application of the SEK joint is in the development of a roller coaster model. This example attempts to replicate the results obtained by Pombo and Ambrósio in their work on roller coasters [57, 58]. Their approach is a significantly higher-fidelity representation of a roller coaster than is developed in this example. The constraints developed by Pombo and Ambrósio allow them to create an accurate representation of the interaction between the track and the wheels of the roller coaster. The model developed using the SEK joint is a simple particle following the trajectory of a roller coaster path; the main goal of this section is to demonstrate an application of the composite path generation algorithm presented in Section 3.2.2.

The roller coaster path from [57] is shown in Figure 4.23. This path is broken into alternating curved and straight segments, as discussed in Section 3.2.2. The x-, y-, and z-components of the track path are shown in Figure 4.24 as a function of the path length, s, after the path generation scheme. The boundaries of each segment are shown using the vertical grey lines. The components of the tangential unit vector are shown in Figure 4.25, and the components of the normal vector are shown in Figure 4.26. Note that the path is C^2 continuous, and that by using the composite path generation procedure, the normal vector is defined along the curve even when the path is straight.

A particle of 1058 kg is constrained to move along the roller coaster path, starting from $s = 0$ and with initial velocity of 2 m/s. This is comparable to the simulation run by Pombo et al. However as mentioned previously, the model developed by Pombo et al. is more detailed and consists of multiple roller coaster cars with wheel bogie systems interacting with a track. Figure 4.27 shows the z-acceleration of the particle as it follows the roller coaster track. It is comparable to the results obtained by Pombo et al., despite the fact that the SEK joint model of the roller coaster was described by a single ODE.

Although a particle was simulated, it is possible to extend the roller coaster model to use a rigid body. In the discussion of the ODE SEK and previous examples, the body orientation was explicitly stated as a function of the path length. However, for a case like the roller coaster, the body orientation is dependent on the shape of the path. This case was discussed for the DAE SEK joint in Section 3.1.4. Recall that the body orientation can be constrained to follow the path, by constructing the following rotation matrix using

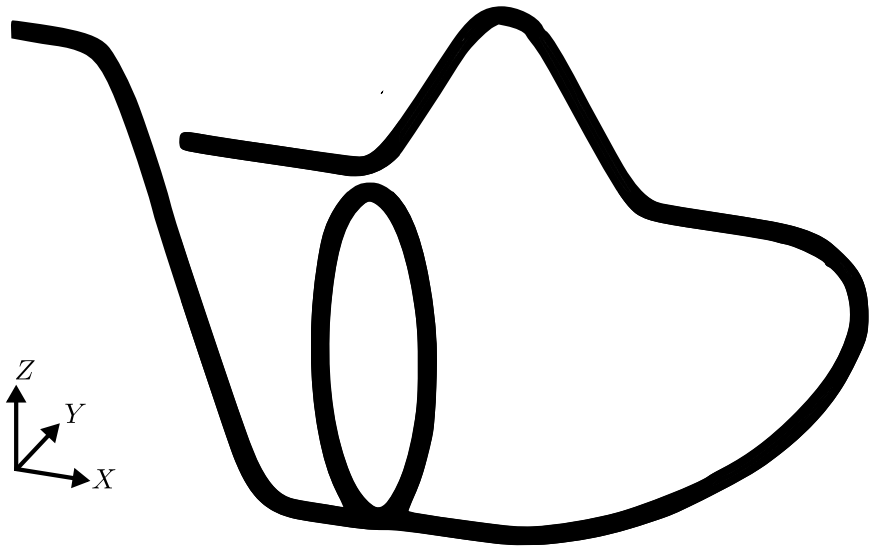


Figure 4.23: Roller coaster path

the Frenet frame unit vectors [66]:

$$\Big[R\Big] = \Big[\{\hat{\mathbf{u}}\} \quad \{\hat{\mathbf{n}}\} \quad \{\hat{\mathbf{b}}\}.\Big] \tag{4.5}$$

To properly define the ODE SEK joint, the angular velocity and acceleration must be symbolically calculated, as well as the change in body orientation with respect to the path length, $\vec{p}(s)$. These equations can be defined as [119]:

$$[\widetilde{\omega}] = \frac{d[R]}{dt}[R]^T \tag{4.6}$$

and

$$[\widetilde{p}] = \frac{\partial[R]}{\partial s}[R]^T. \tag{4.7}$$

The components of $\vec{\omega}$ and $\vec{p}$ can be extracted from $[\widetilde{\omega}]$ and $[\widetilde{p}]$. The cant, or bank, angle of the track can be considered by adding an additional rotation to $[R]$, about the tangential unit vector ($\hat{\mathbf{u}}$). This approach was attempted; however due to the MapleSim implementation limitations discussed in Section 3.4 regarding complex functions in the SEK joint rotation matrix ($[R_{J/I}]$) the simulation was not able to run.

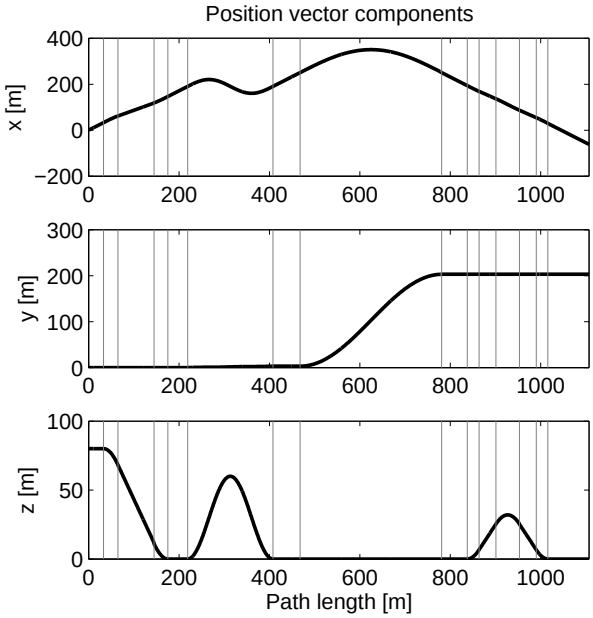


Figure 4.24: Components of the position vector describing the roller coaster path as a function of the path length

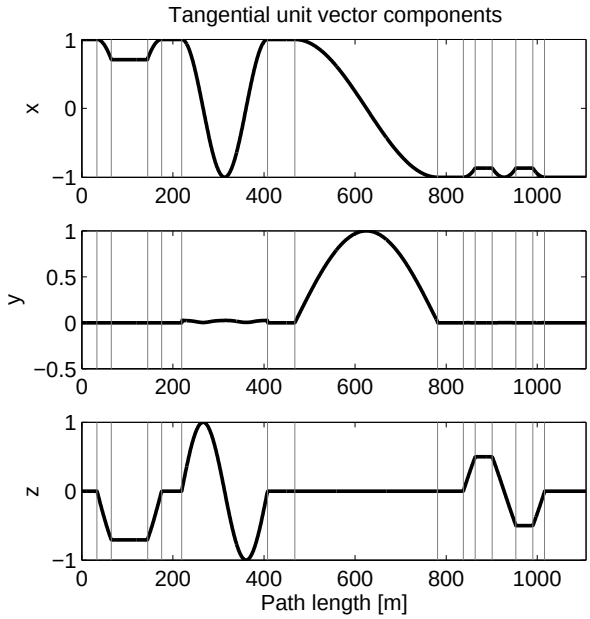


Figure 4.25: Components of the tangential unit vector describing the roller coaster path as a function of the path length

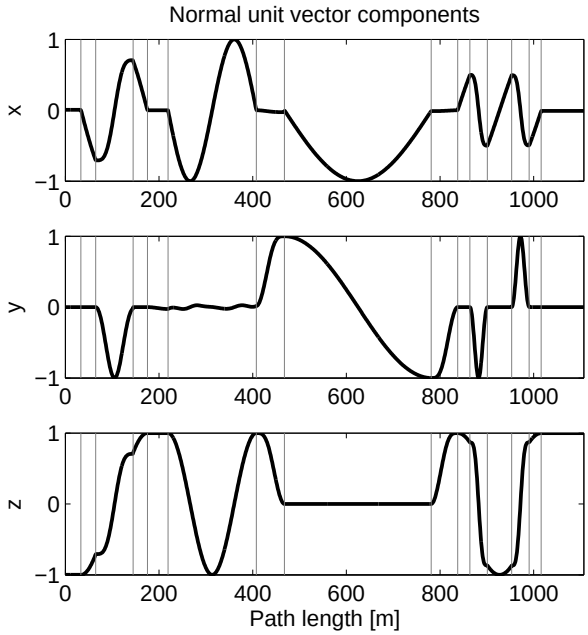


Figure 4.26: Components of the normal unit vector describing the roller coaster path as a function of the path length

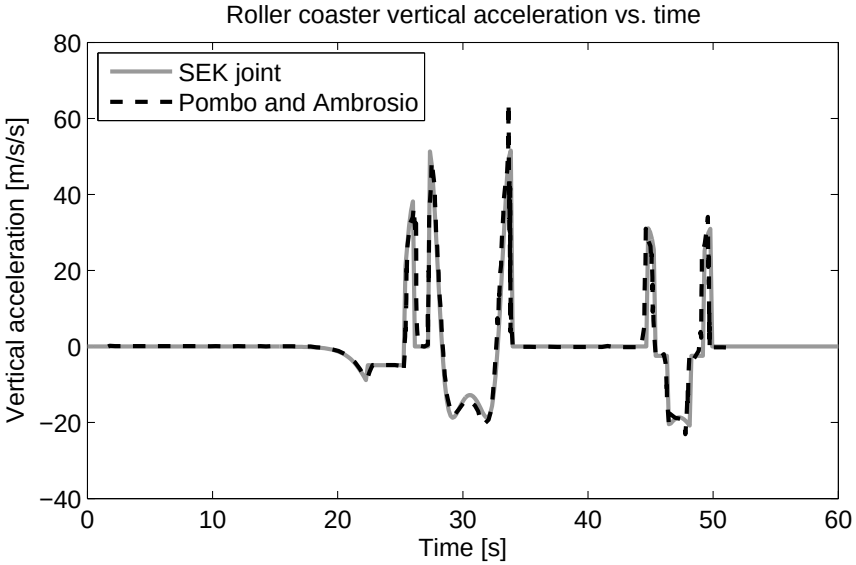


Figure 4.27: Vertical acceleration of the roller coaster model, compared with the results obtained by Pombo and Ambrósio in their work on roller coasters [57, 58]

Chapter 5

Conclusions

In this work, a novel multibody constraint has been created that allows complex mechanisms to be accurately represented using simpler equations than if standard modelling approaches had been used. The resulting Single-DOF Equivalent Kinematic (SEK) joint constrains a body to follow a specific path relative to another body while maintaining a specific orientation. The compliant SEK joint adds compliance properties to the constraint, at the cost of more complex equations. Finally, the Double-DOF Equivalent Kinematic (DEK) joint was created, and extends the SEK joint to constrain a body to follow a surface relative to another body while maintaining a specific orientation. Both versions of the SEK joint and the DEK joint were successfully validated, and examples were provided to show their applicability in models from the mechanical and biomechanical domains.

Two iterations of the SEK joint were developed. The theory of these two joints were presented in Chapter 3. In Section 3.1 the theory of the DAE SEK joint was presented. The theoretical development of this joint was guided by the fact that it was implemented in MapleSim, and so there were constraints placed on the development of the joint. First, a 2D particle joint was developed, before being extended to a full 3D rigid body implementation. The joint was successfully validated against more conventional modelling techniques in both MapleSim and Adams/View. The kinematics of the DAE SEK joint were expressed as a function of the displacement along a Cartesian axis, and so algebraic constraints were introduced to the system, which meant that the dynamics of a kinematic pair were described using a system of DAEs. This meant that one of the main goals of offering a simpler mathematical representation of a system was not met.

A second formulation of the SEK joint was required, which allowed for a simpler mathematical representation. This was presented in Section 3.2. This was realized as the ODE SEK joint and was accomplished by using Frenet frames to describe the mathematics of the constraint. The kinematics of the joint are expressed using splines that are a function of the path length, which helps to eliminate the algebraic constraints and allows complex paths to be generated. To generate the splines and parameterize raw input data in terms of the path length, a path generation scheme was developed. Such an implementation was made possible by developing MapleSim source code, which gave complete freedom over the choice of the formulation. This ODE SEK joint was successfully validated using the previously validated DAE SEK joint.

The ODE SEK joint was then extended to include compliance characteristics. The compliance was modelled as a bushing with stiffness and damping. There is translational compliance perpendicular to the path, and rotational compliance about all three axes. A further extension of the ODE SEK joint was to create the DEK joint, which constrains a body to move along a surface rather than a curve. The DEK joint was a simple extension of the SEK joint, done by increasing the number of coordinates from one to two. A new surface generation scheme was developed so that arbitrary surfaces can be represented using the DEK joint.

In Chapter 4 some practical examples were provided for the SEK and DEK joints. The main example considered the reduction of vehicle suspension models. First, a MacPherson suspension model was created using the SEK joint. This was then extended to a compliant suspension model, before a rigid, steered suspension model was created using the DEK joint. In all cases, the models were compared with equivalent higher-fidelity models developed in MapleSim and Adams/View to show the accuracy of the SEK joint theory as well as the improvements in simulation speed. Finally, a full vehicle model was created using the steered DEK suspension at the front and the rigid SEK suspension at the rear. This model was then compared with an equivalent high-fidelity model using a double lane change maneuver to once again show the strengths of the SEK joint theory.

Two final examples were shown to demonstrate the SEK joint's applicability outside the domain of vehicle dynamics. First, a knee model was made using data from existing literature. No dynamic simulations were performed, but this example demonstrated a potential biomechanical application for the SEK joint. Finally, a roller coaster example was shown. The goal of this example was to demonstrate the composite path generation

scheme that was developed to overcome the limitations of calculating Frenet frame unit vectors for straight lines. This example was compared to a much higher-fidelity roller coaster model from the literature.

5.1 Contributions

There are two significant contributions that the SEK joint offers compared to more traditional modelling approaches. The first is a simpler mathematical representation that enables faster simulation times when compared with high-fidelity models. The second is an alternate representation of a system that can help to reduce the time required to develop a model.

5.1.1 Simpler mathematical representation

By basing the formulation of the SEK joint on Frenet frames and using the path length as a coordinate, all algebraic constraints were removed from the formulation. The simpler mathematical representation allows models to simulate faster when compared to models developed using more conventional approaches. This was demonstrated in all examples shown in Chapter 4.

5.1.2 Alternate topology representation

The concept of a path-following joint also provides an alternative topology representation for many systems. This can reduce the time required to develop models, and also allow models to be developed using less data that would be required for a higher-fidelity model. Considering the examples listed in Chapter 4; each one would require significantly more time and data to construct had a model topology that mimics the physical system been used. For the vehicle suspension example, K&C data can be used to create a SEK joint suspension model; however to generate an accurate high-fidelity model from experimental data, extensive suspension geometry data would be required, before a time consuming correlation procedure is carried out to make sure that the model matched the experimental data. For the example of the knee, the SEK joint provides a simple way to accurately model the kinematics of the knee. Using a more conventional contact model or finite

element approach, extensive information on connective tissue strength and stiffness would be required. Finally, for the roller coaster example, to accurately represent the topology of the physical system a contact model between the car wheels and the track would be required. Alternatively, the SEK joint allows for a model to be developed using only the geometry of the track.

The downside to the alternative topology representation is that detailed information regarding the internal dynamics and forces of a system cannot be computed because they are embedded in the path geometry of the SEK joint. The application of the SEK joint must be carefully considered to ensure that it is used only when detailed information regarding the inner workings of a subsystem is not required.

5.2 Recommendations for future research

The basic theory for the path-following joint has been completely presented in this thesis. However there are still some areas that can be improved, especially with regards to the implementation of the joint.

Improvements to compliance model in compliant SEK joint

Currently the compliance characteristics of the compliant SEK joint are modelled as simple linear bushings with constant stiffness along the path. A simple extension of this would be to include a non-linear compliance model in the SEK joint that can vary along the length of the path. This means that (3.91) to (3.97) could be rewritten as arbitrary functions of s:

$$F_n = f_n(n, \dot{n}, s) \tag{5.1}$$

$$F_b = f_b(b, \dot{b}, s) \tag{5.2}$$

$$M_\theta = f_\theta(\theta, \dot{\theta}, s) \tag{5.3}$$

$$M_\phi = f_\phi(\phi, \dot{\phi}, s) \tag{5.4}$$

$$M_\psi = f_\psi(\psi, \dot{\psi}, s) \tag{5.5}$$

Surface spline generation algorithm for DEK joint

Currently the surface generation algorithm is quite primitive and can only generate a surface polynomial. This means that the DEK joint can only be used to model kinematic pairs where the relative motion surface is representable by a single polynomial. Extending the surface generation scheme to have the same generality of the SEK joint path generation algorithm would greatly increase the versatility of the DEK joint.

Non-uniform rational B-spline (NURBS) surfaces are commonly used to represent complex surfaces [120,121]. The main problem with B-splines is their recursive definition, which makes them time consuming to solve [89]. This is why in the path generation scheme the final step in the pre-processing scheme was to convert the B-splines to simple piecewise polynomials. Some researchers have proposed non-recursive methods to represent NURBS surfaces explicitly [122]; however these methods have been shown to be unreliable [123] and have not achieved widespread attention or use. Further research should be undertaken to explore the feasibility of using NURBS surfaces in the DEK joint.

Compliant DEK joint

A compliant DEK joint could be easily created by extending the rigid DEK joint presented in this thesis. This would follow a similar path to the work that was done to derive the compliant SEK joint from the rigid SEK joint. The additional joint coordinates, and compliance bushing forces and moments would need to be introduced to the joint formulation. This type of joint would be useful for modelling compliant steered suspension systems as well as 2-DOF biological joints.

Removal of limitations on size of rotation matrix entries

As discussed in Section 3.4, Maple struggles to simulate SEK joint models in which the rotation matrix entries are large splines. This future work would require close collaboration with Maplesoft to see if the SEK joint equations coded into MapleSim can be manipulated to reduce their complexity so that the MapleSim equation building routines are able to formulate the dynamic equations without errors.

Improved implementation for the end user

An eventual goal for the SEK and DEK joints is for other researchers, and even Maplesoft customers, to use the joints in their models. Currently this is not possible given the ad-hoc implementation method of overwriting existing MapleSim library components. New components should be added to the MapleSim library that allow a user to interact with all of the SEK joint parameters in an easy way using the MapleSim user interface.

The path and surface generation algorithms are implemented in Matlab, and are currently not suitable for use by end users of MapleSim. Implementation of these algorithms in Maple, with a user-friendly interface, is also required before other users could take advantage of the SEK and DEK joints. Such an algorithm would allow the user to import their kinematic data for the SEK or DEK joint into MapleSim and then Maple routines could automatically generate the paths for each SEK or DEK joint in the model.

References

[1] MSC Software Corp. *ADAMS/Car 2010 Technical Manual*, 2010.

[2] W. F. Milliken and D. L. Milliken. *Race Car Vehicle Dynamics*. Society of Automotive Engineers, Warrendale, 1995.

[3] W. F. Milliken, D. L. Milliken, and M. Olley. *Chassis design: principles and analysis*. Society of Automotive Engineers Warrendale, PA, 2002.

[4] D. A. Crolla and R. P. King. Olley's "flat ride" revisited. In *The dynamics of vehicles on roads and on tracks, Supplement to vehicle system dynamics, Volume 33. Proceedings of the 16th IAVSD symposium held in Pretoria, South Africa, August 30-September 3, 1999*, 2000.

[5] R. S. Sharp. Wheelbase filtering and automobile suspension tuning for minimizing motions in pitch. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 216(12):933–946, 2002.

[6] A. M. C. Odhams and D. Cebon. An analysis of ride coupling in automobile suspensions. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 220(8):1041–1061, 2006.

[7] N. Orlandea and M. A. Chace. Simulation of a vehicle suspension with the Adams computer program. Technical Report 770053, SAE, 1977.

[8] W. Kortüm. Review of multibody computer codes for vehicle system dynamics. *Vehicle System Dynamics*, 22(S1):3–31, 1993.

[9] D. Cao, X. Song, and M. Ahmadian. Editors perspectives: Road vehicle suspension design, dynamics, and control. *Vehicle System Dynamics*, 49(1-2):3–28, 2011.

[10] H. Schuette and P. Waeltermann. Hardware-in-the-loop testing of vehicle dynamics controllers–a technical survey. Technical Report 2005-01-1660, SAE, 2005.

[11] T. Bach. Real-time simulation in anti-lock brake system development based on a personal computer. In *VDI 6th International Congress "Measurement and Testing Techniques in the Automotive Industry"*. VDI, 1992.

[12] A. Maciel, L. P. Nedel, and C. M. D. S. Freitas. Anatomy-based joint models for virtual human skeletons. In *Computer Animation, 2002. Proceedings of*, pages 220–224. IEEE, 2002.

[13] M. Machado, P. Flores, J. C. P. Claro, J. Ambrósio, M. Silva, A. Completo, and H. M. Lankarani. Development of a planar multibody model of the human knee joint. *Nonlinear Dynamics*, 60(3):459–478, 2010.

[14] MSC Software Corporation. Adams, the multibody dynamics simulation solution. `http://www.mscsoftware.com/product/adams`, 2013. Retrieved 9/3/2013.

[15] M. W. Sayers. *Symbolic computer methods to automatically formulate vehicle simulation codes*. PhD thesis, University of Michigan, 1990.

[16] M. W. Sayers. Vehicle models for RTS applications. *Vehicle System Dynamics*, 32 (4–5):421–438, 1999.

[17] M. W. Sayers and D. Han. A generic multibody vehicle model for simulating handling and braking. *Vehicle System Dynamics*, 25(1):599–613, 1996.

[18] T. D. Gillespie. *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers, Warrendale, 1992.

[19] R. N. Jazar. *Vehicle dynamics: Theory and application*. Springer, New York, 2008.

[20] F. Jindra. Mathematical model of four-wheeled vehicle for hybrid computer vehicle handling program. Technical Report DOT HS 801800, National highway traffic safety administration, 1976.

[21] R. W. Allen, H. T. Szostak, D. H. Klyde, T. J. Rosenthal, and K. J. Owens. Vehicle dynamic stability and rollover. Technical Report DOT HS 807956, National highway traffic safety administration, 1992.

[22] Mechanical Simulation Corp. *CarSim Reference Manual*, 6.05 edition, 2006.

[23] J. Tobolar. The virtual axle–a simplified vehicle suspension model for real-time applications. *Computational Mechanics*, 2:461–468, 2002.

[24] C. W. Kim, H. K. Jung, J. S. Shim, and C. W. Kim. Development of vehicle dynamics model for real-time electronic control unit evaluation system using kinematic and compliance test data. *International Journal of Automotive Technology*, 6(6):599–605, 2005.

[25] N. Biggs, E. K. Lloyd, and R. J. Wilson. *Graph Theory, 1736-1936*. Clarendon Press, 1986.

[26] J. J. McPhee, C. Schmitke, and S. Redmond. Dynamic modelling of mechatronic multibody systems with symbolic computing and linear graph theory. *Mathematical and Computer Modelling*, 10(1):1–23, 2004.

[27] J. J. McPhee. On the use of linear graph theory in multibody system dynamics. *Nonlinear Dynamics*, 9(1):73–90, 1996.

[28] T. S. Dao, A. N. Seaman, and J. J. McPhee. Mathematics-based modeling of a series-hybrid electric vehicle. In *5th Asian Conference on Multibody Dynamics 2010*, 2010.

[29] J . M. Banerjee and J. J. McPhee. Symbolic sensitivity analysis of multibody systems. *Multibody Dynamics*, pages 123–146, 2013.

[30] C. Schmitke, K. Morency, and J. J. McPhee. Using graph theory and symbolic computing to generate efficient models for multi-body vehicle dynamics. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, 222(4):339–352, 2008.

[31] N. Gachadoit and R. Renaud. Modeling and design of an active suspension system with maple and maplesim. In *Mechatronics, 2012 9th France-Japan & 7th Europe-Asia Congress on and Research and Education in Mechatronics (REM), 2012 13th Int'l Workshop on*, pages 425–432. IEEE, 2012.

[32] E. Yam. Topology-based vehicle systems modelling. Master's thesis, University of Waterloo, 2012.

[33] W. Rulka and E. Pankiewicz. MBS approach to generate equations of motions for HiL-simulations in vehicle dynamics. *Multibody System Dynamics*, 14(3/4):367–386, 2005.

[34] A. Eichberger. Generating multibody real-time models for hardware-in-the-loop applicationsdynamics. *ATA-TORINO-*, 56(5/6):164–167, 2003.

[35] A. Eichberger and W. Rulka. Process save reduction by macro joint approach: the key to real-time and efficient vehicle simulation. *Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility*, 41(5):401–413, 2004.

[36] M. Acevedo and J. T. Celigüta. Real time dynamic simulation of passenger cars. *Mechanics of Structures and Machines, Proc. Mechatronics and Supercomputing Applications in the Transportation*, pages 559–566, 1994.

[37] C. R. Carlson and J. C. Gerdes. Optimal rollover prevention with steer by wire and differential braking. In *Proceedings of IMECE*, pages 16–21, 2003.

[38] T. Shim and C. Ghike. Understanding the limitations of different vehicle models for roll dynamics studies. *Vehicle System Dynamics*, 45(3):191–216, 2007.

[39] T. Shim and D. Toomey. Investigation of active steering/wheel torque control at the rollover limit maneuver. *SAE Transactions*, 113(6):1133–1140, 2004.

[40] T. Shim and D. Margolis. Dynamic normal force control for vehicle stability enhancement. *International Journal of Vehicle Autonomous Systems*, 3(1):1–14, 2005.

[41] W. Drozdz and H. B. Pacejka. Development and validation of a bond graph handling model of an automobile. *Journal of the Franklin Institute*, 328(5):941–957, 1991.

[42] T. D. Day, S. G. Roberts, and A. R. York. Simon: a new vehicle simulation model for vehicle design and safety research. Technical Report 2001-01-0503, SAE, 2001.

[43] J. Song. Performance evaluation of a hybrid electric brake system with a sliding mode controller. *Mechatronics*, 15(3):339–358, 2005.

[44] J. He, D. A. Crolla, M. C. Levesley, and W. J. Manning. Integrated active steering and variable torque distribution control for improving vehicle handling and stability. *SAE Transactions*, 113(6):638–647, 2004.

[45] N. Cooper, D. Crolla, M. Levesley, and W. Manning. Integration of active suspension and active driveline to ensure stability while improving vehicle dynamics. Technical Report 2005-01-0414, SAE, 2005.

[46] C. Kim and P. I. Ro. An accurate full car ride model using model reducing techniques. *Journal of Mechanical Design*, 124(4):697–705, 2002.

[47] Y. Ando and M. Suzuki. Control of active suspension systems using the singular perturbation method. *Control Engineering Practice*, 4(3):287–293, 1996.

[48] E. S. Kim. Nonlinear indirect adaptive control of a quarter car active suspension. In *Control Applications, 1996., Proceedings of the 1996 IEEE International Conference on*, pages 61–66. IEEE, 1996.

[49] J. Rauh. Virtual development of ride and handling characteristics for advanced passenger cars. *Vehicle System Dynamics*, 40(1-3):135–155, 2003.

[50] P. Morse. Using K&C measurements for practical suspension tuning and development. Technical Report 2004-01-3547, SAE, 2004.

[51] W. C. Mitchell, R. Simons, T. Sutherland, and M. Keena-Levin. Suspension geometry: Theory vs. K&C measurement. Technical Report 2008-01-2948, SAE, 2008.

[52] P. S. Rao, D. Roccaforte, R. Campbell, and H. Zhou. Developing an Adams model of an automobile using test data. Technical Report 2002-01-1567, SAE, 2002.

[53] A. Hall and J. J. McPhee. Automation of Adams/Car K&C correlation using MATLAB. Technical Report 2014-01-0847, SAE, 2014.

[54] Mechanical Dynamics Inc. *ADAMS/Rail 9.1 Technical Manual*, 1995.

[55] Intec GmbH. *SIMPACK Reference: SIMPACK Element Catalogue*, 2001.

[56] K. Schott and J. Tobolar. Spatial tracks in multibody simulation. theory and simple roller coaster example. *Proceedings in Applied Mathematics and Mechanics*, 3(1): 162–162, 2003.

[57] J. Pombo and J. Ambrósio. General spatial curve joint for rail guided vehicles: Kinematics and dynamics. *Multibody System Dynamics*, 9(3):237–264, 2003.

[58] J. Pombo and J. Ambrósio. Modelling tracks for roller coaster dynamics. *International Journal of Vehicle Design*, 45(4):470–500, 2007.

[59] J. Ambrósio. Trainset kinematic: a planar analysis program for the study of the gangway insertion points. Technical Report IDMEC/CPM/97/005, Institute of Mechanical Engineering, Instituto Superior Tecnico, Lisbon, Portugal, 1997.

[60] H. Akima. A new method of interpolation and smooth curve fitting based on local procedures. *Association for Computing Machinery*, 17(4):589–602, 1970.

[61] C. De Boor. *A practical guide to splines*. Springer, 1978.

[62] Larry D Irvine, Samuel P Marin, and Philip W Smith. Constrained interpolation and smoothing. *Constructive Approximation*, 2(1):129–151, 1986.

[63] C. A. Micchelli, P. W. Smith, J. Swetits, and J. D. Ward. Constrained lp approximation. *Constructive Approximation*, 1(1):93–102, 1985.

[64] R. L. Bishop. There is more than one way to frame a curve. *American Mathematical Monthly*, pages 246–251, 1975.

[65] M. Tändl, A. Kecskeméthy, and M. Schneider. A design environment for industrial roller coasters. In *CD Proceedings of the ECCOMAS Thematic Conference on Advances in Computational Multibody Dynamics,(Milano, Italy)*, 2007.

[66] A. Kecskeméthy and M. Tändl. A robust model for 3d tracking in object-oriented multibody systems based on singularity-free frenet framing. *Advances in Robot Kinematics*, pages 255–264, 2006.

[67] E. Andersson, M. Berg, and S. Stichel. *Rail Vehicle Dynamics: Fundamentals and Guidelines*. Railway Technology, 1999.

[68] I. A. Kapandji. The physiology of the joints. *Edinburgh: Churchill Livingstone*, 1974.

[69] W. Shao and V. Ng-Thow-Hing. A general joint component framework for realistic articulation in human characters. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 11–18. ACM, 2003.

[70] S. L. Delp, J. P. Loan, M. G. Hoy, F. E. Zajac, E. L. Topp, and J. M. Rosen. An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *Biomedical Engineering, IEEE Transactions on*, 37(8):757–767, 1990.

[71] S. Lee and D. Terzopoulos. Spline joints for multibody dynamics. *ACM Trans. Graph*, 27(3):1–8, 2008.

[72] R. M. Murray, Z. Li, and S. S. Sastry. *A mathematical introduction to robotic manipulation*. CRC, 1994.

[73] Y. Fan. Multi-body dynamics: Spline joint, 2012.

[74] J. Park and P. Nikravesh. Effect of steering-housing rubber bushings on the handling responses of a vehicle. In *SAE International Congress & Exposition, Detroit, Michigan*, 1997.

[75] M. V. Blundell. The influence of rubber bush compliance on vehicle suspension movement. *Materials & Design*, 19(1):29–37, 1998.

[76] J. Ambrósio and P. Verissimo. Improved bushing models for general multibody systems and vehicle dynamics. *Multibody System Dynamics*, 22(4):341–365, 2009.

[77] Morse Measurements LLC. Summary of standard K&C tests and reported results. `http://www.morsemeasurements.com/technical/description-of-kc-tests/`, 2011. Retrieved 24/2/2013.

[78] MTS Systems Corporation. Dynamic kinematic and compliance testing. `http://www.mts.com/ucm/groups/public/documents/library/dev_002222.pdf`, 2011. Retrieved 4/3/2013.

[79] E. Meijering. A chronology of interpolation: from ancient astronomy to modern signal and image processing. *Proceedings of the IEEE*, 90(3):319–342, 2002.

[80] O. Neugebauer. *Astronomical Cuneiform Texts: Babylonian Ephemerides of the Seleucid Period for the Motion of the Sun, the Moon, and the Planets:[In 3 Vol.]*, volume 1. Published for the Institute for Advanced Study, 1955.

[81] C. C. Gillispie. *Dictionary of scientific biography*. Scribner, 1981.

[82] L. Yan and D. Shiran. Chinese mathematics: A concise history. *AMC*, 10:12, 1987.

[83] H. H. Goldstine. *A History of Numerical Analysis from the 16th through the 19th Century*. Springer, 1977.

[84] J. Karup. Über eine neue mechanische ausgleichungsmethode. In *Transactions of the Second International Actuarial Congress*, pages 31–77. London, UK: Layton, 1899.

[85] G. King. Notes on summation formulas of graduation, with certain new formulas for consideration. *Journal of the Institute of Actuaries*, 41(4):530–565, 1907.

[86] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. part a–on the problem of smoothing or graduation. a first class of analytic approximation formulae. *Quart. Appl. Math*, 4(1):45–99, 1946.

[87] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. part b–on the problem of osculatory interpolation. a second class of analytic approximation formulae. *Quart. Appl. Math*, 4(2):112–141, 1946.

[88] K. Branson. A practical review of uniform b-splines, 2004.

[89] L. L. Schumaker. *Spline functions: basic theory*, volume 1981. Wiley New York, 1981.

[90] D. L. B. Jupp. Approximation to data by splines with free knots. *SIAM Journal on Numerical Analysis*, 15(2):328–343, 1978.

[91] T. M. Tao and A. T. Watson. An adaptive algorithm for fitting with splines. *AIChE journal*, 34(10):1722–1725, 1988.

[92] G. E. Hölzle. Knot placement for piecewise polynomial approximation of curves. *Computer-Aided Design*, 15(5):295–296, 1983.

[93] M. J. Lindstrom. Penalized estimation of free-knot splines. *Journal of Computational and Graphical Statistics*, 8(2):333–352, 1999.

[94] B. Su and D. Liu. *Computational geometry: curve and surface modeling*. Academic Press Professional, Inc., 1989.

[95] W. Li, S. Xu, G. Zhao, and L. P. Goh. Adaptive knot placement in b-spline curve approximation. *Computer-Aided Design*, 37(8):791–797, 2005.

[96] G. Mamic and M. Bennamoun. Automatic bayesian knot placement for spline fitting. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 169–172. IEEE, 2001.

[97] S. H. Lane, J. W. Lee, and T. A. Watson. An algorithm for determining smooth continuous pressure derivatives from well-test data. *SPE formation evaluation*, 6 (04):493–499, 1991.

[98] Vehicle Dynamics Standards Committee. Vehicle dynamics terminology. Technical Report J670e, SAE, 1976.

[99] C. Schmitke, Director, MapleSim Development, Maplesoft. Personal communication, 2012-2014.

[100] E. J. Haug. *Computer aided kinematics and dynamics of mechanical systems*, volume 1. Allyn and Bacon Boston, 1989.

[101] MSC Software Corp. *ADAMS/View 2010 Technical Manual*, 2010.

[102] J. H. Ginsberg. *Advanced Engineering Dynamics*. Cambridge University Press, 1998.

[103] P. R. Halmos. *Naive set theory*. Springer, 1998.

[104] H. Wang, J. Kearney, and K. Atkinson. Arc-length parameterized spline curves for real-time simulation. In *5th International Conference on Curves and Surfaces. Oslo, Norway*, 2002.

[105] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107(3):391–408, 2006.

[106] E. Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, New Jersey, 2007. 827 pp.

[107] H. Goldstein. *Classical Mechanics*, volume 4. Addison-Wesley, Massachusetts, 1962.

[108] K. Kuczynski. Carpometacarpal joint of the human thumb. *Journal of Anatomy*, 118(Pt 1):119, 1974.

[109] F. J. Valero-Cuevas, M. E. Johanson, and J. D. Towles. Towards a realistic biomechanical model of the thumb: the choice of kinematic description may be more critical than the solution method or the variability/uncertainty of musculoskeletal parameters. *Journal of biomechanics*, 36(7):1019–1030, 2003.

[110] S. Gupta, F. C. T. Van Der Helm, J. C. Sterk, F. Van Keulen, and B. L. Kaptein. Development and experimental validation of a three-dimensional finite element model of the human scapula. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 218(2):127–142, 2004.

[111] C. Högfors, B. Peterson, G. Sigholm, and P. Herberts. Biomechanical model of the human shoulder jointii. the shoulder rhythm. *Journal of biomechanics*, 24(8):699–709, 1991.

[112] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 11(2):431–441, 1963.

[113] MSC Software Corporation. Angular results from output requests have unexpected jumps. Online knowledge base article : KB8012409, 2014. Retrieved 23/5/2014.

[114] A. Hall, T. Uchida, F. Loh, C. Schmitke, and J. J. McPhee. Reduction of a vehicle multibody dynamic model using homotopy optimization. *Archive of Mechanical Engineering*, 60(1):23–25, 2013.

[115] Mechanical Dynamics Inc. *ADAMS/Pre 11.0 Reference Guide*, 2001.

[116] J. Angeles. *Rational kinematics*, volume 34. Springer, 1988.

[117] MSC Software Corp. *ADAMS/Car 2013 Technical Manual*, 2013.

[118] L. E. DeFrate, R. Papannagari, T. J. Gill, J. M. Moses, N. P. Pathare, and G. Li. The 6 degrees of freedom kinematics of the knee after anterior cruciate ligament deficiency an in vivo imaging analysis. *The American journal of sports medicine*, 34 (8):1240–1246, 2006.

[119] D. Greenwood. *Principles of Dynamics, the 2nd Edition*. Englewood Cliffs, NJ Prentice-Hall, 1988.

[120] D. F. Rogers and R. Earnshaw. *State of the art in computer graphics: visualization and modeling*. Springer, 1991. 225–269 pp.

[121] L. Piegl. On nurbs: a survey. *IEEE Computer Graphics and Applications*, 11(1): 55–71, 1991.

[122] L. Liu and G. Wang. Explicit matrix representation for nurbs curves and surfaces. *Computer Aided Geometric Design*, 19(6):409–419, 2002.

[123] D. O. M. Alacid. A shape sensitivity analysis module with geometric representation by nurbs for a 2d finite element program based on cartesian meshes independent of the geometry. Master's thesis, Universitat Politècnica De València, 2012.

APPENDICES

Appendix A

Derivations

A.1 2D particle dynamic equation derivation

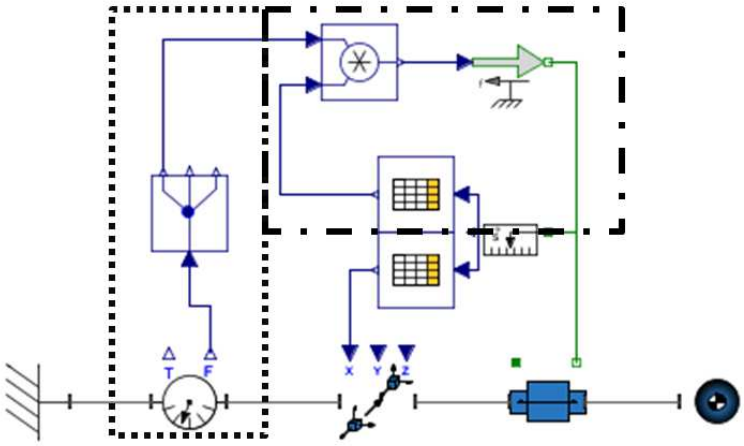


Figure A.1: Topology of 2D particle SEK joint

Figure A.2 shows the symbolic DAEs generated by MapleSim for the 2D particle SEK joint (Figure A.1), obtained using the supplied Equation Extraction template. Equations (1) and (3) are the ODEs and equations (2), (4) and (5) are the AEs. Equation (2) is the simplest to understand, it simply states that the x-location of the particle is determined using the lookup table, with the input as the z-displacement of the prismatic joint, essentially $x = \mathbf{f}(z)$. Similarly, equation (4) defines $x' = \mathbf{f}(z)$. Equation (5) is the calculation of the

$$-2 + 3\left(\frac{d}{dt}\left(\frac{d}{dt}\,Zmotion\_s(t)\right)\right) - Zmotion\_F(t) = 0 \quad (1)$$

$$X(t) = \text{`dsn/Interp1Dsim`}(1, 2, Zmotion\_s(t), 0) \quad (2)$$

$$\frac{d}{dt}\left(\frac{d}{dt}X(t)\right) = -\frac{1}{3}P4\_u1(t) + \frac{5}{3} \quad (3)$$

$$XslopeFuncZ\_y\_1(t) = \text{`dsn/Interp1Dsim`}(1, 3, Zmotion\_s(t), 0) \quad (4)$$

$$Zmotion\_F(t) = P4\_u1(t)\,XslopeFuncZ\_y\_1(t) \quad (5)$$

Figure A.2: MapleSim equations for 2D particle SEK joint

$$Main.AWF3.AFInputX.Fx(t) - 3\left(\frac{d^2}{dt^2}Main.PT1.PTInputX.X(t)\right) \quad (6)$$

$$0 \quad (7)$$

$$Main.AWF3.AFInputZ.Fz(t) - 3\left(\frac{d^2}{dt^2}Main.Zmotion.s(t)\right) \quad (8)$$

Figure A.3: `GetInterFrameForce` output for 2D particle SEK joint

z-component reaction force, F_{N_z}. Using the Multibody Analysis template and the Maple command `GetInterFrameForce` the equations generated by the force moment sensor can be seen (Figure A.3). The symbolic equation defining the variable $P4\_u1(t)$ is equation (6), which is the result of the force sensor and demultiplexer shown in the dotted rectangle in Figure A.1. Equation six describes this relationship:

$$P4\_u1(t) = F_{A_x}(t) - ma_x(t) \quad (A.1)$$

meaning that $P4\_u1(t)$ is F_{N_x}. Therefore equation (5) reduces to

$$F_{N_x}\left(\frac{\partial x}{\partial z}\right) = F_{N_x}\frac{F_{N_z}}{F_{N_x}} = F_{N_z}. \quad (A.2)$$

Consequently, equation (1) is equivalent to

$$\sum F_z : F_{A_z} + F_{N_z} = ma_z \quad (A.3)$$

where $m = 3$ kg and $F_{A_z} = 2$ N. Finally, equation (3) is clearly equivalent to

$$\sum F_x : F_{A_x} + F_{N_x} = ma_x \tag{A.4}$$

where $F_{A_x} = 1$ N.