2010 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
MODELING & SIMULATION, TESTING AND VALIDATION (MSTV) MINI-SYMPOSIUM
AUGUST 17-19 DEARBORN, MICHIGAN

A Framework for Modeling and Simulation of Tracked Vehicles of High Mobility

Peilin Song, PhD **Pete Melick** **James Horchner**

Mobility, Power and Energy Branch
AMSAA, APG, MD

ABSTRACT

This paper presents a modeling and simulation framework for tracked vehicles for ride comfort and load prediction analysis. The development began with the identification of the key issues such as formulations, integration schemes and contact (with friction) modeling on which the comparative studies are conducted. Based on the results of the investigations, the framework and process for the modeling and simulation of tracked vehicles are established and appropriate algorithms for contact and friction are developed. To facilitate the modeling and simulation process, a Python-based modeling environment was developed for process automation, design optimization and design of experiment. The developed framework has been successfully applied to the dynamic load predication of a M1A1 based Joint Assault Bridge (JAB). The parameter optimization enabled with the Python-based process automation tool helps improve the design and modification of vehicles for significantly improved fatigue life of suspension component.

INTRODUCTION

A tracked-vehicle is a vehicle that runs on the continuous track instead of wheels. The track mechanisms in the vehicle rotate the looped tracks to provide an endless patch of contact surfaces between the track, road-wheel and surface so the mobility of the vehicle is increased significantly, especially on the soft-soil, low friction and uneven grounds such as mud, ice and snow.

However, the increased mobility comes with a price: the complexity of the track mechanisms in comparison to the wheels. The complexity not only pertains to the design and manufacturing aspects of the track mechanism, but also the inherent difficulty to understand the mechanical behaviors which determine performance, reliability and fatigue life. Classical analytical methods fall short in predicting the characteristics of the systems with such level of complexity and the application of traditional test approaches is also limited because of the daunting cost and severity of the test environments. As an alternative, physics-based simulation is sought as a virtual test tool for performance prediction and evaluation.

Multi-body simulation programs have been used in the simulation of tracked vehicles and a certain level of success has been achieved. However, due to the maturity level of simulation technologies and the short history of user experiences, fast, reliable, and robust simulations are hard to achieve for applications like the load prediction of the track on the vehicle. This paper creates a framework of a track simulation environment for the modeling and simulation of tracked vehicles for the high fidelity simulations that produce dynamic load prediction required for reliability and fatigue analysis. In the following chapters, we will first present an overview of this framework, then discuss the key issues and components and, last, we present some results from successful simulations.

The work presented in this paper is focused on the continuous track made of a number of rigid units that are joined to each other. However, other kinds of tracks such as flexible belt can also be incorporated with minimal change.

MODELING AND SIMULATION FRAMEWORK FOR TRACKED VEHICLES

Vehicular tracks are featured by a large number of parts connected together which are in contact with the ground. It exerts the following challenges to the modeling and simulation practices:

It is difficult to build the model because of their large number of parts, connections and forces. Some commercial analysis programs come with the template-based tools for the vehicle modeling such as ADAMS/ATV[4], RecurDyn/Track Toolkit[3] and have been used in the applications. However, the flexibility is compromised because of the encapsulation of model building algorithms. Flexibility to support a new type of track model and advanced simulations such as design of experiments are hard to realize.

There are no robust and efficient contact algorithms to model the contact between tracks and sprockets and between the track pads and ground. General contact algorithms supplied by the analysis software results in lengthy simulations, and can cause the simulation to fail unexpectedly. It is hard to adjust the contact parameters for speedy and fast simulation and it is also impossible to incorporate advanced features such as traction and resistance predictions which are central to the mobility analysis and fuel consumption prediction.

There is no consensus on the formulations, algorithms and integration schemes for the tracked vehicle simulations, and the selection of “right” tools are, to some extent, influenced by the software vendor rather than technical merits.

In this paper, we will present our understanding and investigation results on the issues of concerns and propose a framework for the modeling and simulation of tracked vehicles. This framework includes:

1. Dedicated contact models defined to support the contacts between the track pad and ground (in mesh form). In addition to ensuring reliable, fast and robust simulations, the models can be also customized to support a variety of soils in contact for mobility, dynamic ride quality and dynamic load prediction.
2. Tracked vehicle modeling and simulation automation environments. The environment is written in Python and is independent of any commercial modeling tools. The tracked vehicle modeling is driven by the parameters such as road wheel station positions and the dependencies defining the topology and dimensions are resolved internally. The modeling elements such as parts, connections and forces are automatically created and output into files for use by commercial simulation programs.

The proposed modeling and simulation framework has been successfully used in the dynamic load prediction of a M1A1-based Joint Assault Bridge (JAB) and has demonstrated tangible improvement in load predictions that ultimately were utilized to extend component fatigue life.

SOME ISSUES OF CONCERNS

In this chapter, two issues are put forward and discussed in depth:

1. Formulations of tracked vehicles
2. Integration schemes for tracked vehicle simulation

A physical system can be perceived differently from the modeling and simulation perspective for different purposes. The different views on the physical systems directly determine the formulation and integration scheme selections. For the tracked vehicle with high mobility demand, high fidelity simulations are required. There are two distinctive features:

1. The interactions between the track links are defined by elastic forces instead of revolute connection;
2. The interactions between the link pad and ground are defined by contact force rather than kinematic relationship.

Those two features increase the size of a track model significantly, increase the simulation time and require the robust integration schemes.

On the Formulation of Tracked Mechanism

Traditional wisdom and intuitions believe that since a track is composed a number of links serially connected, recursive formulation is the first of the choices. Recursive formulation uses the relative between the adjacent links and calculates the positions, velocities and accelerations from bottom up and force from top down. The advantage of a recursive method can result in small and well-conditioned equation set if and only if applied appropriately; i.e., the configuration of a link is determined kinematically with the link closer to the root (to ground). In the model for high mobility track, there is no kinematic joint connecting the adjacent links so there is no advantage to take from the recursive formulations. Thus, Cartesian coordinate formulation is the better choice.

In multi-body dynamics programs, the implicit integration schemes prevail. The performance of an implicit integration scheme is mainly determined by the integration Jacobian and its calculation speed and quality. For the modeling and simulation of the track with high mobility, the Cartesian coordinate method will generate much smaller Jacobian of higher quality. For example, a track of 50 links contains 50*6=300 degrees of freedoms. In the recursive formulations, its contribution to the Jacobian is a dense matrix of 300*300 (900000 non-zeros) while in the Cartesian coordinate method, it is defined by a block-tridiagonal matrix of 300*300 with 5328 non-zeros. The ratio of non-zeros in the Jacobians between the two formulations is 5328/900000=0.0592. The Cartesian coordinate formulation is more suitable for the modeling of the tracked with high mobility.

On the Integration Schemes for Tracked Vehicle Simulation

The models of the track with high mobility are featured with high stiffness connection between links and high stiffness contact between track pads and the ground. These forces are not only large in number, but also their natures of discontinuity and high stiffness which generate the high frequency dynamics in the simulations. The models contain both low frequency dynamics for large motions and the high frequency dynamics for the components participating in the contacts. The co-existence of high and low frequency dynamics leads to a set of very stiff nonlinear equations that makes the Backward Differential Formula (BDF) integrators run very slow since they are trying to address both low and high frequency dynamics.

In addition to the physical high frequency dynamics, the parasitic high-frequency dynamics also exists as the by-product of contact modeling. Those high frequency dynamics are of little interest from a user's perspective; it is a residual effect and does not affect the global motion of systems under modeling. To achieve a fast simulation with reasonable accuracy for global motion of low frequency, an integrator able to apply numerical damping for high frequency dynamics is desired.

Hilber Hughes Taylor (HHT) integrator (together with its variants such as alpha-method and beta method) is a desired and well-proved method to deal with the models with residual high frequency dynamics. Those integration schemes employ various interpolation methods to filter out the high frequency dynamics while the low frequency dynamics are well preserved [2]. The numerical dumping introduced in the HHT integrator results in the energy dissipation of order O(h^2) which is much more pronounced for high frequency dynamics than low-frequency dynamics [5].

The HHT integration scheme has many years of successful history with ABAQUS and MARC and is available in some multi-body analysis programs such as ADAMS and RecurDyn. For example, the simulation time of a Crowler track model in ADAMS for 14 seconds has been cut from 8988 seconds to 1713 seconds as reported with ADAMS 2005[2].

ADAMS is the only software the authors are aware of that satisfies the requirements for formulations and integrations. It was adopted in our framework as the primary tool.

MODEL AND ALGORITHMS FOR THE PAD AND GROUND CONTACT AND FRICTIONS

The most significant feature in the track modeling is the contact forces between the link pad and ground. The contacts not only support the vehicle from falling, but also provide the traction/brake force to push/stop the vehicle. A good model and algorithm for the contact dictate the quality requirement of track model and simulation. "Good" model here has both physical and numerical meanings: in addition to reflecting the physical natures of the contacts, it should also enable fast and accurate simulations. In this chapter, we will study both contact force and traction force modeling issues separately and propose suitable modeling practices.

Contact Force Formulation

The contact force is based on the penalty function with respect to the penetration of the pad into the ground. It is the method suggested in ADAMS and widely in use.

F=K(Dy-Dy0)^n+CVy

In which Dy-Dy0 is the penetration and F is zero when the penetration is zero or negative.

In our implementation, we pinpoint 4 points on the contact surface of the pad and measure the penetration and velocity. The value of the stiffness, K, and n, the soil ground softness exponent, are determined from the test data. However, we have found, contrary to intuition and previous arguments [1], that the value of damping has little effects on the convergence, or simulation speed, for global motion within the range 0.01 > C/K > 0 .001. The plausible reason could be that, for the high frequency dynamics, the numerical damping applied by the HHT integrator overshadows the physical damping in effects. This phenomenon is only observed with hard surface contact and the experience with soft soil ground is not available at present.

Traction/Friction Force Formulation

Friction between the pad and ground provides the traction force to the vehicle. Due to the complex nature of friction and its significance in the propulsion, the modeling of the friction is the most import yet weakest link in the vehicle track modeling. It is the largest contributing factor to the convergence, simulation speed, and the accuracy of the simulation because the micro-motion (slip) associated with the contact in the direction of the friction directly affects the global motion in longitudinal direction.

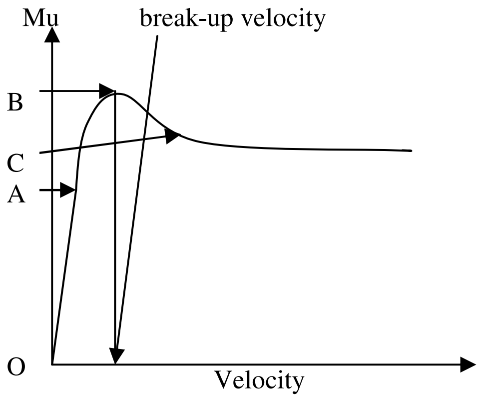

Figure 1 Friction Force Coefficient vs. Relative Velocity

Friction is a function of relative velocity between the surfaces in contact and is built up during the contact in three stages: Elastic phase, transition phase and sliding phase. They can be illustrated in Figure 1.

Initially, friction is mainly due to elastic deformation of pad and/or ground surface. In this phase, the relationship between the force and relative velocity is linear. This corresponds to section OA of the curve. A further increase of the driving force results in part of the pad sliding on the ground until the stiction breaks at point B. Under these circumstances, the relationship between the force and velocity is nonlinear. The sections OA and AB combined forms the static friction regime. After point B where the break-up occurs, the motion is of 100% sliding and section BC represents the transition from static friction regime to dynamic friction regime. The relationship OABC in Figure 1 is embedded into the contact model and the friction force is represented as the function of velocity and contact force.

This friction model contains no stiffness in its formulation, which means the poles associated with the friction are on the imaginary axis. The numerical stability of this model cannot be guaranteed in the vicinity of zero velocity where the linear deformation of pad and/or ground dominated. The numerical difficulty encountered without stiffness defined is demonstrated by the long time to complete simulations and inability of the model to converge.

Adding the stiffness in the vicinity of zero velocity to the friction model is suggested. In addition to the numerical stability improvement, the fidelity of model is also improved since the elastic deformation dominates the traction efforts in the phase one of the contact. As such, a force of the stiffness is defined in the friction model, which is a function of both stiffness and displacement. During the simulation, both forces, velocity dependent and displacement dependent, are computed at the same time and the displacement dependent force supersedes the velocity dependent force if the former is below the force at break-up (point B in Figure 1). Otherwise, the force determination falls into velocity regime.

Traditionally, there are concerns about this approach in that the high stiffness introduced by the addition of the stiffness to the friction model may deteriorate the simulation performance [1].

This used to be the case with BDF integrators. However, the introduction of HHT integrator, which dampens the high frequency vibrations, makes this case invalid.

The addition of stiffness in the friction model in the vicinity of zero velocity increases the robustness and speed of the simulation significantly. With this implementation, we are able to simulate a M1A1 JAB (with 512 parts and 9000+ forces in the model) running on an uneven terrain for 20,000 feet in only 6 hours on a 32-bit computer with dual core processors.

MODELING ENVIRONMENT FOR TRACKED

Model build, modification and update of complicated systems are tedious works and also prone to error. Frequent model creation, modification and update are needed in applications to answer what-if questions. It is especially true for the modeling and simulations of systems used in the parameter optimization and design of experiments (DOE) for which the automated model creation and update are demanded so they can be hooked in the automated process.

A dynamic model of a system is a collection of model data and the relationship and dependency between the data. However, the dependency and relationship between the data are static rather than dynamic: the relationship and dependency are frozen during the model creation and any subsequent model update requires the users to reevaluate the data relationship and dependency. While some commercial software such as ADAMS can parameterize the models, it is difficult to incorporate the algorithms into the product to resolve the dependency and update the parameters. The lack of the tools to provide both capability and flexibility for model creation, and modification is the motivation behind the creation of a neutral environment for model creation and update.

The modeling environment is actually a Python program which can be used to create, store and output the modeling data. Central to this environment are two components: the database for storing the modeling data and the algorithm to create, modify and update the data by sorting out the data relationship and resolving the dependency. In the implementation, Objected Oriented Programming paradigm is applied so the data and algorithms are encapsulated in classes. The topology and hierarchy in the modeling data are inherently represented in the class definition

```
class Model
        |
        |---class Part
        |        |-- class marker
        |---class Joint
        |
        |--class Force
        |
        |--class subsystem
                |---class Part
                |        |--class marker
                |--class Joint
                |
                |--class Force
```

Figure 2 Class Hierarchy of the Building Block in the Modeling Environment

Each class contains the relevant modeling data and the algorithm for the relationship definition and dependency. As shown in the Figure 3, in additional to the top level Model class and primitives such as Part, Joint and Force, a subsystem class is also defined to support the encapsulation of the data at different levels. For example, the tracked vehicles can be instanced incorporating the subsystems.

```
TrackedVehicle (class Model)
        |
        |---Chassis (class Part)
        |---LeftSprocket (class Part)
        |---RightSprocket(class Part)
        |---LeftTrack (class sybsystems)
                |--link1 (class Part)
                :
                |--contactForce1 (class Force)
                :
        |---RightTrack (class sybsystems)
                |--link1 (class Part)
                :
                |--contactForce1 (class Force)
```

Figure 3 Instantiation of Tracked Vehicle Class in the Model

The instantiation of a tracked vehicle is also completed in a Python program. Since all the data and algorithms are encapsulated in the classes, the –tracked vehicle model can be easily created, updated and parameterized with the different parameters passed to the Python program. The model stored in Python database can be exported to the files which can be read by different CAE software. For the time being, only MSC.ADAMS is supported. However, support to other formats is not excluded.

The flexibility and power of Python programming language is further applied in the environment to support following features:

1. Simulation automation and parameter optimizations

   The model creation and update can be conducted automatically and successively so different optimization schemes can be integrated into the model creation and simulation process to realize the model optimization.

2. Customization of the User Subroutines for Contact and Friction Forces

   The algorithms for contact and friction forces are coded in forms of user function and the discrete representation of 3D ground terrain are a tedious work. Python script is used to read the terrain data from files and put it directly into the user-defined function. This ensures the full automation of model creation.

APPLICATIONS AND CONCLUSIONS

The framework proposed for the modeling and simulation of tracked vehicles has been successfully applied in the prediction of dynamic ride quality and dynamic load for several variants of the M1A1-based JAB. The track vehicle model built in MSC.ADAMS has been verified and validated. A model of two tracks is shown below with chassis invisible. This model contains 512 parts and over 9000 forces and is among the largest models in multi-body dynamic simulations. With the customized contact and friction models embedded in user functions, we are able to run the simulation of this tracked vehicle running on an uneven terrain for 20,000 feet (with simulation time 150 seconds) in only 6 hours on a 32-bit computer with dual core processor. A frame of simulation is shown in Figure 4. The ease of model creation and modification, the fast, robust and accurate simulation and Python-based automation tool are the technology enablers with which multiple runs on each variant on different terrains can be competed in a reasonably short time and DOE-based parameter optimization is feasible. The applications of this framework in modeling, simulation and parameter optimization onto the M1A1-based JAB was used to support suspension optimization that resulted in a tangible increase in the fatigue life of some components exposed to frequent contact and impact. The framework proposed and presented in this paper is not limited to the tracked model modeling and simulation and can be extended to other types of vehicles and mechanical systems.

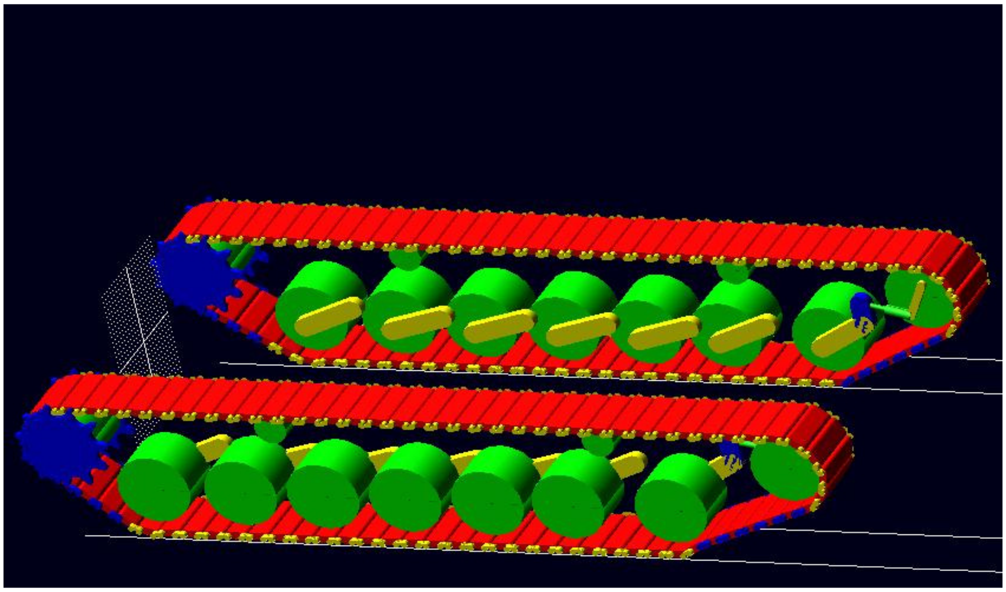

Figure 4 Graphical Representation of Track Model in MSC.ADAMS

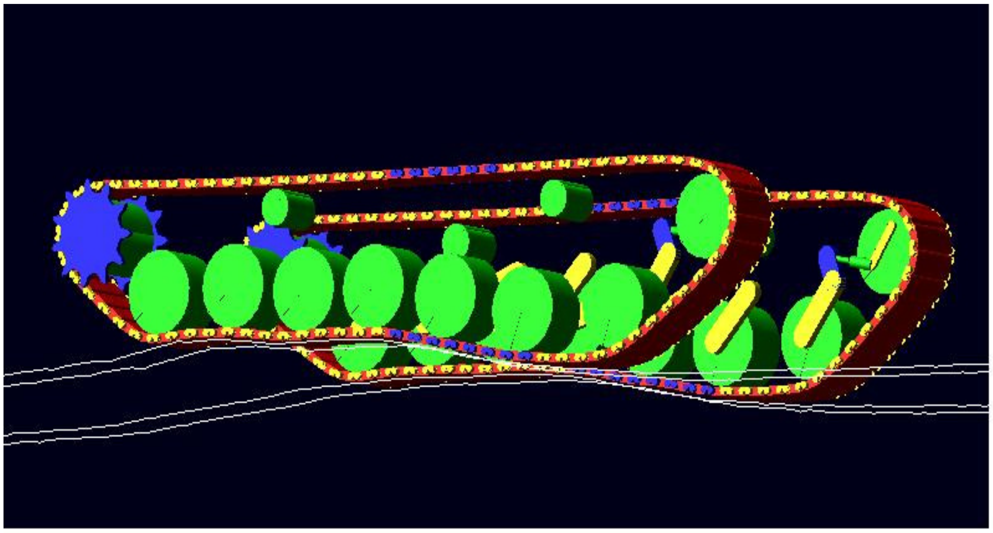

Figure 5 A Snapshot of Track Model Running on Uneven Terrain

At AMSAA, the applications of this framework also successfully create the dynamic models of military automotive platforms. At present, this effort supports the modeling and simulation of vehicles for mobility and fuel consumption analysis on soft soil.

REFERENCES

1. J. Slattengren, Simulation of Complex Systems Dominated by Contact and Friction, 1999, MSC KB Article 8014627
2. D. Negrut, R. Rampalli, G. Ottarsson, and A. Sajdak, On an Implementation of the HHT Method in the Context of Index 3 Differential Algebraic Equations of Multibody Dynamics, ASME Journal of Computational and Nonlinear Dynamics, Vol. 2, No. 1, pp. 73-85, 2007.
3. B. Ross, RecurDyn Motion Simulation Software Simulates Track Loader, Desktop Engineering, March 4, 2009
4. L. Zeferino, Simulation of Tracked Vehicles at Giat Industries using ADAMS and ATV, 16th European MDI User Conference, Nov. 2001
5. D. Negrut, L. Jay, N, Khude, A Discussion of Low-Order Numerical Integration Formulas for Rigid and Flexible Multibody Dynamics, J. Computational Nonlinear Dynamics April 2009, Volume 4, Issue 2