2013 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
MODELING & SIMULATION, TESTING AND VALIDATION (MSTV) MINI-SYMPOSIUM
AUGUST 21-22, 2013 - TROY, MICHIGAN

REAL-TIME AND HIGH-FIDELITY SIMULATION ENVIRONMENT FOR AUTONOMOUS GROUND VEHICLE DYNAMICS

Jonathan Cameron, Ph.D.
Steven Myint
Calvin Kuo
Abhi Jain, Ph.D.
Håvard Grip, Ph.D.
Jet Propulsion Laboratory
California Institute of Technology

Paramsothy Jayakumar, Ph.D.
U.S. Army TARDEC

Jim Overholt, Ph.D.
U.S. Air Force Research Laboratory

ABSTRACT

Integrated simulation capabilities that are high-fidelity, fast, and have scalable architecture are essential to support autonomous vehicle design and performance assessment for the U.S. Army's growing use of unmanned ground vehicles (UGV). The HMMWV simulation described in this paper embodies key features of the real vehicle, including a complex suspension and steering dynamics, wheel-soil models, navigation, and control. This research uses advanced multibody techniques such as minimal coordinate representations with constraint embedding to model complex unmanned ground vehicles for fast mechanical simulations with high fidelity. In this work, we demonstrate high-fidelity dynamics models for autonomous UGV simulations in near real time that can be useful to the U.S. Army for future autonomous ground vehicle dynamics modeling and analysis research.

INTRODUCTION

With increased onboard autonomy, advanced vehicle models are needed to analyze and optimize control design and sensor packages over range of urban and off-road scenarios. Moreover, integrated simulation capabilities that are high-fidelity, fast, and have scalable architecture are essential to support autonomous vehicle design and performance assessment for the U.S. Army's growing use of unmanned ground vehicles.

Recent work at TARDEC has attempted to develop a high-fidelity mobility simulation of an autonomous vehicle in an off-read scenario using integrated sensor, controller, and multi-body dynamics models [1]. The conclusion was that (a) real-time simulation was not feasible due to the complexity of the intervening formulation, (b) models had to be simplified to speed up the simulation, (c) interfacing the sensors was exceedingly difficult due to co-simulation, (d) the controls developed were very basic and could not be optimized, and (e) a rigid terrain model was used.

The JPL ROAMS ground vehicle simulation framework is based on the JPL Darts/Dshell simulation architecture [2], [3], [4], [5]. ROAMS and the underlying architecture have been successfully demonstrated at JPL in several space mission-critical scenarios where a high degree of mission complexity, real-time performance, and extensive sensor, actuator, and control integration were necessary. The underlying framework has been applied to a variety of mission-critical simulation needs for NASA missions across multiple domains (cruise/orbiter, landers, and rovers) over the years. The architecture has supported real-time embedded hardware-in-the-loop use to large-scale Monte Carlo based parametric studies. For further information about ROAMS and vehicle modeling at JPL, see http://dartslab.jpl.nasa.gov.

ROAMS is unique in its integrated approach to straddling the multi-function high-fidelity dynamics, sensors, environment, control, and autonomy models that are key attributes of future Army unmanned ground vehicles (UGVs).

The project described in this paper was a pilot effort to demonstrate the application of the ROAMS modeling approach for addressing the fidelity and speed bottlenecks for TARDEC and the Army's needs for the evaluation and testing of autonomous ground vehicles. The simulation architecture represents a shift in paradigm in empowering analysts with full visibility and control in tailoring key elements of the simulator. This scalable architecture allows the adaptation and tuning of simulation fidelity across a very broad range (e.g. rigid/flex-body dynamics, sensor fidelity, dynamics/kinematics modes) needed for the multi-layered testing of complex autonomy behaviors. This feature is in contrast with alternative approaches that focus on specific aspects such as sensor fidelity, vehicle dynamics, or behavioral models that address only a narrow slice of vehicle autonomy simulation needs. This simulation approach has been successfully used by analysts across multiple NASA centers for a variety of problems.

Leveraging prior work done at JPL for autonomous planetary rovers, the authors have adapted the JPL's ROAMS vehicle simulation framework to develop vehicle models with the following attributes: multibody dynamics based on the fast recursive order-N spatial algebra formulation, wheeled locomotion capability, vehicle model based on a common military vehicle (the HMMWV), selected models of sensors (cameras, GPS, radar, LIDAR), and actuators, off-road compliant terrains with Bekker/Terazaghi soil interaction models [3], drive-to-goal locomotion planning with obstacle avoidance, drive/steer to maintain the vehicle stable while following a prescribed path and avoiding obstacles, and 3D real-time graphics visualization and data logging capability.

The simulator's architecture will allow the seamless selection of different fidelity levels and model parameters across the full modeling suite, and more importantly, provide analysts with a modular way to swap component models for varying vehicle/control/sensor behavior.

MODELING THE HMMWV SUSPENSION SYSTEM

The HMMWV suspension system is a variant of the common double wishbone suspension. These suspension systems have a large number of distinct bodies that are contained within a single kinematic closed loop. As a result, these suspensions have a large number of internal degrees of freedom, but due to the constraints imposed by them to a frame or chassis, they only have a single independent degree of freedom[1].

There are two issues that arise from constrained systems in computational dynamics modeling. The first is the added computation required to satisfy the constraints on the system. The second is the error drift that results over time from the discretized integration of the multibody dynamics equations. This error drift is usually handled by error correction algorithms that are imposed after each discretized step to minimize the constraint error over time, adding even more computation for each constraint.

For the HMMWV suspension modeling, we have tested and benchmarked three algorithmic techniques to solve the multibody dynamics of the suspension system. While the HMMWV suspension model is the same, the difference between the three techniques is the number of constraints that are needed, which directly affects their resultant computational speed. These techniques are described in the following paragraphs as well as a later section; for more detail, see [6].

The first technique is known as the fully augmented (FA) technique and models each body in the system as fully independent (with six degrees of freedom) and then imposes constraints between the rigid bodies to simulate the connectivity of the system. This results in a large number of constraints, however the dynamics computation is relatively simple due to the assumption that all of the bodies are independent. The computations involved are order $O(N^3)$ where N is the number of bodies.

The second technique is known as the tree augmented (TA) technique where constraints are only imposed in closed chains. This is done by taking the multibody system and converting it to a tree topology by "cutting" any closed chains and imposing a constraint at the cut. The rest of the tree topology system is then modeled using minimal coordinates. This results in constraints only appearing in closed chains, however the minimal coordinates computation is significantly more difficult than in the FA case. For TA, the computations involved are order $O(N)$.

The final technique is known as the constraint embedded (CE) technique which converts all closed chains into variable shape compound bodies which have the same number of degrees of freedom as the number of independent degrees of freedom for the closed chain systems they replace. These compound bodies locally handle their internal degrees of freedom and constraint, effectively hiding them from the

[1] *This assumes that the suspension joints are represented as pin joints. When explicitly modeled, bushings introduce additional degrees of freedom.*

dynamics solver. As a result, the CE method has no constraints. For CE, the computations involved are order O(N).

In general for small closed chain systems embedded in a multibody system, the CE technique is significantly faster than the TA or the FA technique. This is because the CE technique, which hides the constraints in compound bodies, is able to directly use the fast recursive algorithms for tree systems.

We used a standalone suspension model for one wheel (this is sometimes called a “quarter-car” model) to develop CE models and benchmark its performance against the TA and FA approaches. See Figure 1.

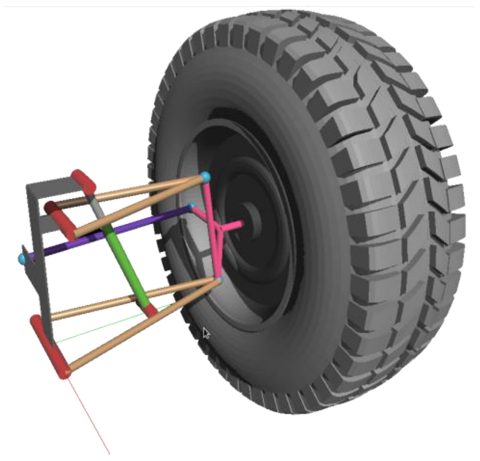
Figure 1: Single standalone HMMWV “quarter-car” wheel suspension model

The quarter-car model includes the double “A-arm” closed-chain mechanism and the associated spring-damper unit. We tested the quarter-car model by running a simulation by applying a known force profile to the wheel over a period of several seconds. We tested and benchmarked the suspension model by using the three techniques for modeling the closed chain mechanism described previously. All three approaches produced the same motions, providing a degree of cross-validation.

The CE technique proved to be both the fastest and have the least constraint error over time. The next fastest was the TA method, followed by the non-analytic CE method. Normally, the CE technique solves the internal system using an iterative approach; however we eventually developed an analytic solver for the quarter-car suspension system, which further reduced the dynamics computation time. Adding bushings may require different algorithms for the closed-loop calculations. The parameters for the suspension system were determined from an ADAMS model provided by the sponsor [1].

To integrate the suspension model into the full HMMWV model, the rear suspensions were simply attached to the chassis. However, the front suspensions required an additional steering mechanism, which was modeled as a four-bar closed chain to which the suspension tie rod was connected. The result was a coupled steering system that uses the steer angle as a single independent degree of freedom in the four-bar which in turn steers both suspension systems.

HMMWV Vehicle Parameters and Graphical Representation

The kinematic and mass properties for this HMMWV simulation were taken from an ADAMS model developed by [7] and provided to us by the task sponsor. The CAD models for the graphical representation were also taken from the same ADAMS model. The wheel-soil interaction was modeled using parameters similar to those in past ROAMS planetary rover simulations and are representative of moderately firm sandy soil [3].

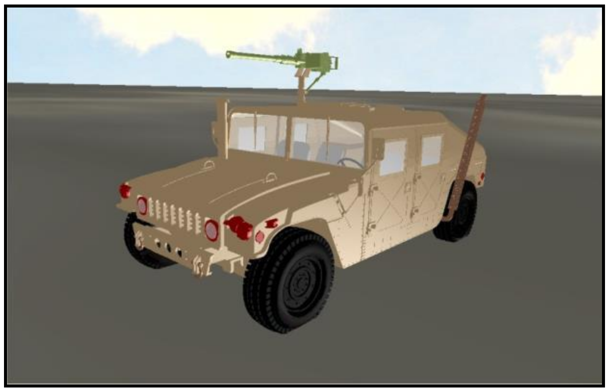
Figure 2: HMMWV Simulation Model

Closed-Chain Suspension Modeling

In this section we outline some of the mechanism modeling advances [6] that were used to construct the vehicle simulation for this task.

Here we consider the dynamics of multibody systems with closed-loop topologies such as in Figure 3. The relative motion between adjacent bodies in a multibody system is constrained. There are several approaches to handle systems with internal closed loop constraints. Such inter-body constraints may be treated as hinges, or as closure constraints between the bodies. In the hinge approach, the constraint is eliminated from the equations of motion by using minimal relative coordinates to parameterize the allowable motion between the body pair. On the other hand, the closure constraint approach employs non-minimal coordinates and utilizes bilateral constraints and Lagrange multipliers to enforce the restricted motion between body pairs.

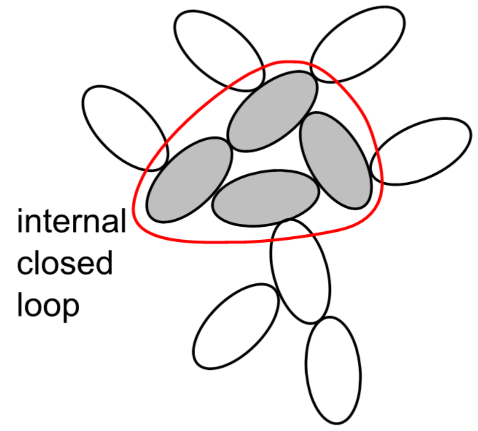

Figure 3: A closed-chain multibody system with an internal closed loop constraint

While the use of hinges leads to equations of motion with smaller dimension, the accompanying minimal coordinates exhibit a high degree of dynamical coupling and a more complex formulation. However, the added complexity can be mitigated by the use of structure-based, low order, recursive methods for the dynamics computations. In contrast, the exclusive use of closure constraints leads to larger, but structurally simpler forms of the equations of motion. More generally, closed-chain models contain a hybrid combination of hinges and closure constraints, with the hinges being associated with a tree topology sub-system within the overall model. The following three approaches for closed-chain dynamics span the range of these modeling options:

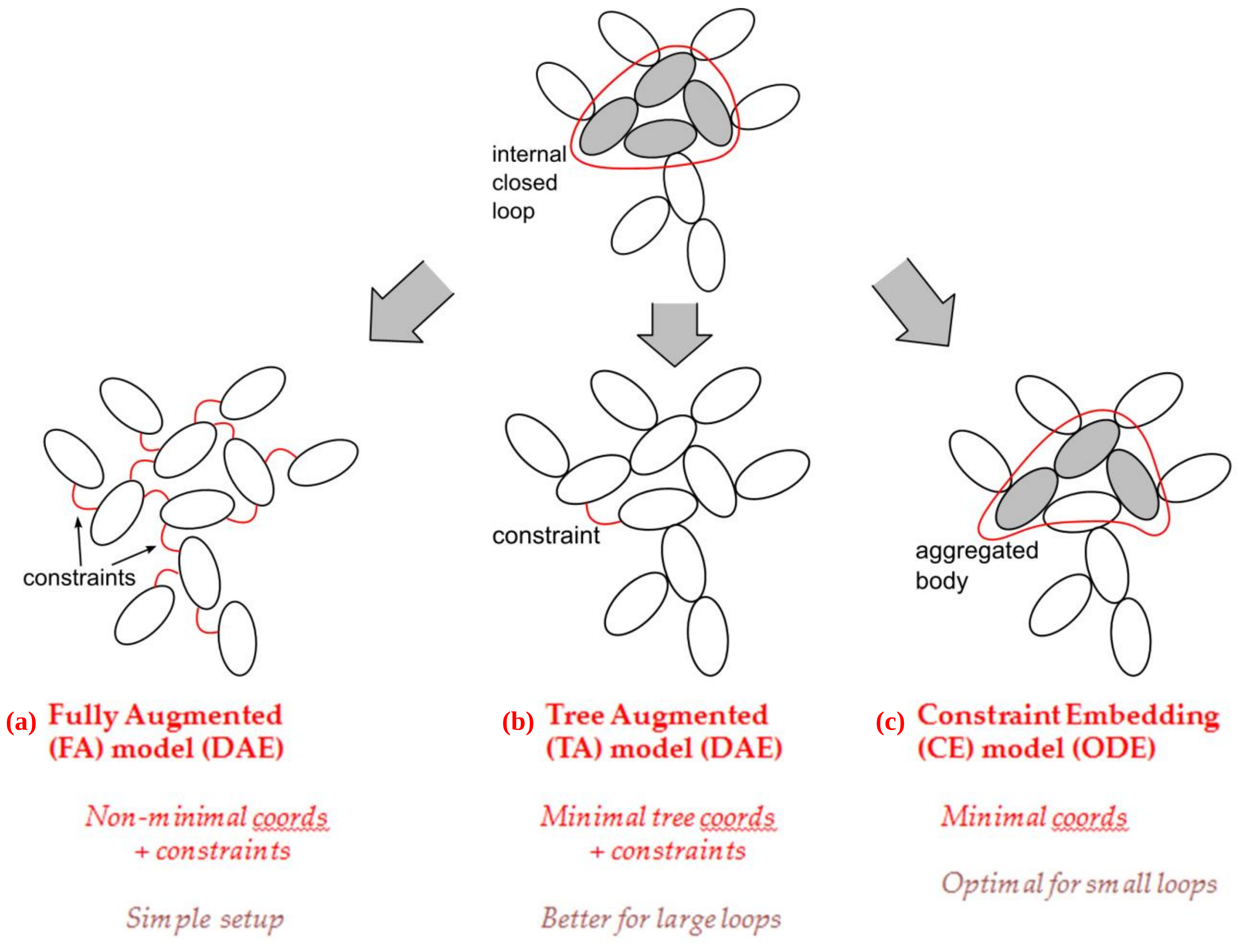

Figure 4: In the fully augmented model (FA), all bodies are treated as independent bodies with inter-body constraints. In the tree augmented model (TA), the system is decomposed into a tree system together with a minimal set of inter-body constraints. In the constraint embedding model (CE), internal loops are aggregated into bodies to convert the system into a tree topology system.

Fully-augmented (FA) method: The first method is the fully-augmented (FA) dynamics method. In this method, all bodies are modeled as independent bodies using absolute coordinates, and the restricted relative motion is modeled via closure constraints as shown in Figure 4(a). The advantage of this approach is that the equations of motion are simple and easy to set up. The mass matrix of the tree sub-system is block diagonal and constant. Sparse matrix solution techniques can be used to solve for Lagrange multiplier constraint forces. Disadvantages include the use of a large number of non-minimal generalized coordinates, the underlying differential-algebraic equation (DAE) structure of the equations of motion, and the need for error control techniques to manage constraint error growth during a simulation.

Tree-augmented (TA) method: The second method is the tree-augmented (TA) dynamics method. In this method, a minimal set of the inter-body constraints are "cut" to obtain the tree-topology sub-system. The maximal spanning tree based tree sub-system only has hinges as illustrated in Figure 4(b). The overall dynamics model consists of the minimal-coordinate dynamics model for the tree sub-system together with a minimal set of closure constraints. The number of generalized coordinates and closure constraints is much smaller compared with the FA model. The tree system's mass matrix however is dense and configuration dependent, though low-order recursive algorithms are available for solving the tree system dynamics without requiring mass matrix inversion. The underlying formulation remains a DAE, but constraint error control is only needed for the smaller set of closure constraints.

Constraint embedding (CE) method: The third method is the new constraint embedding (CE) dynamics method. This technique uses the TA model as a starting point and its closure constraints are eliminated by aggregating bodies affected by the closure constraint into compound bodies as shown in Figure 4(c). The transformed system has a tree topology with only inter-body hinges and no closure constraints. The benefit of this minimal coordinates approach is that low-order tree algorithms can be directly used to solve the dynamics. Also, this formulation results in an ordinary differential equation (ODE) instead of a DAE, and constraint error control techniques are not required. This method however is more complex to implement, since the aggregated bodies now contain multiple rigid bodies and have configuration dependent geometry. While the CE method shares the minimal coordinates attribute with projection dynamics techniques, its advantage lies in the preservation of the system's tree topology that is necessary for the use of the structure-based tree algorithms.

Dynamic Model Performance

The HMMWV vehicle model built in ROAMS has 15 degrees of freedom (DOF) after taking into account all the constraints of the system. In addition to the constraint embedding (CE) HMMWV vehicle model, we also implemented tree augmented (TA) and fully augmented (FA) models of the vehicle. These three modeling approaches were evaluated in terms of speed by simulating the suspended HMMWV subjected to external disturbances, such as those from the terrain. The table below lists the number of coordinates required by the different approaches, as well as their speed performance.

Method	No. of coord-inates	No. of constr-aints	Augment-ed size	Sim time ratio
CE	15	0	15	1
TA	45	30	75	4.1
FA	216	201	417	120.0

As described earlier, the FA model treats all bodies as independent rigid bodies and treats all hinges as bilateral constraints between the bodies. The FA model has 216 coordinates with the size of the bilateral constraints being 201. As expected the difference between them is 15, which is the number of essential independent coordinates for the system. However the size of the equations of motion is determined by the sum of these numbers which works out to 417 for the FA model. The TA model uses minimal coordinates for the spanning tree of bodies in the model, together with a minimal set of bilateral constraints for the joint cuts used to obtain the spanning tree. The size of this model is much smaller with 45 tree coordinates and bilateral constraints of size 30. The CE model uses constraint embedding to eliminate even these remaining constraints in the TA model. The size of the CE model is the minimal 15 size and there are no additional bilateral constraints. The last column in the table compares the relative simulation time for each of these models for the same time step size and simulation duration. The TA model is 4 times slower, while the FA model is 120 times slower. These short benchmarking runs did not include constraint error control for the TA and FA model runs that would otherwise be required for the longer runs.

ENVIRONMENT

The vehicle model was tested in two main environments: an urban environment and an off-road environment. In both cases a graphical representation of the environment was created and used to form a digital elevation map for use in the vehicle wheel-terrain contact simulation. The graphical representation, as rendered in OpenGL, was also used by the LIDAR sensor.

Urban Environment

The environment consisted of a 3D mesh model of a city. The mesh was created using the commercial tool CityEngine [8], which generates realistic urban environments that can be exported to 3D mesh files.

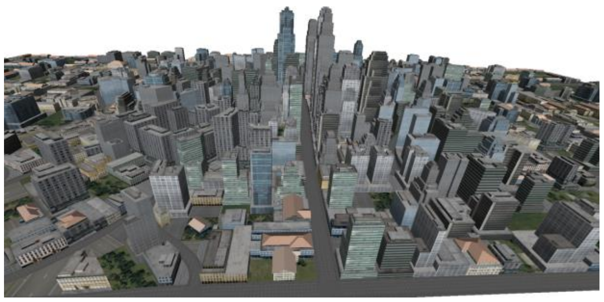

Figure 5: Urban environment created by CityEngine

The digital elevation map required for simulating the vehicle's interaction with the ground was extracted from the CityEngine-generated meshes.

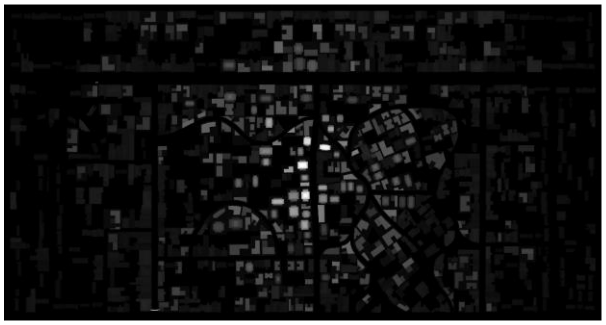

Figure 6: Depth map for urban environment created by CityEngine

Off-Road Environments

We also simulated the vehicle on off-road environments, consisting of bumpy terrain, as opposed to the paved flat urban environment. The off-road digital elevation map was generated using the Height Map Editor tool [9], which generates random elevation maps with natural-looking hilly terrains.

Figure 7: Off-road environment

SENSORS

The LIDAR was simulated by rendering the 3D scene in OpenGL and reading back the depth buffer in a raster that matched the raster characteristics of a typical LIDAR unit. The depth values were treated as a simulated flash LIDAR output.

The simulation system uses the SPICE [10] system to determine relative frames of objects with respect to planetary bodies. This capability was used to simulate GPS coordinate lookup.

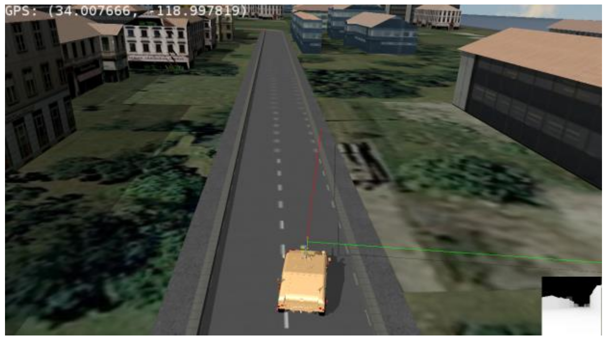

Figure 8: Simulation showing simulated LIDAR (bottom right) and GPS output (top-left)

The view out the driver's window that is shown in Scenario 1 shows the camera sensor used in ROAMS simulations. The HMMWV simulation did not include stereo cameras; however, this capability exists in the ROAMS framework and can be used to construct depth maps for navigational purposes.

AUTONOMOUS DRIVING AND OBSTACLE AVOIDANCE

In addition to modeling the HMMWV, we also developed a simple autonomous driving routine for navigating through obstacles in an environment. This autonomous system has two parts: sensing and planning. Sensing is used to locate obstacles in the world while planning is used to find a path through the obstacles to the eventual destination. It should be noted that both the sensing and planning can be swapped out for more sophisticated autonomy algorithms and models, but this was outside the scope of this project.

The sensing algorithm detects obstacles by directly querying the terrain DEM (digital elevation map). The algorithm works by asking the terrain for the absolute height at various coordinates and calculating the relative height with respect to the HMMWV's current ground plane. If the relative height is above a certain threshold, it is marked as an obstacle and its footprint is expanded in the internal HMMWV map as a safety measure. While the current sensing model uses truth data not normally accessible by a real world system, it is possible to utilize data from a simulated LIDAR image to create similar obstacle maps. The LIDAR is currently in the form of a grid of range values, which can easily be interpreted as a point cloud. The terrain modeling software has procedures to take point clouds and convert them to its native digital elevation map representation, which can be used for vehicle simulation.

The motion planning algorithm is based on an arc planner that draws a number of arcs from the HMMWV's current location. The planner checks each arc to see which are clear of obstacles, and selects the obstacle-free arc that brings the HMMWV closest to its goal. The arcs are recalculated at a set delta time ΔT. This planning algorithm can easily be swapped out an alternate algorithm such as a D* path planning algorithm.

SIMULATION PLATFORM

The ROAMS HMMWV vehicle simulation runs on a Linux computer running the Fedora Core operating system (version 17). Most of the software is written in C++ in order to run efficiently as possible. However, Python is used extensively during simulation setup to construct the C++ parts, connect them together and install the user-specified parameters to configure the simulation. Visualization is done using custom 3D graphics software based on OGRE and OpenGL. The software runs on a standard Linux computer.

SCENARIOS

We consider three scenarios that illustrate the various goals of the project.

Scenario 1 - Urban driving with navigation and obstacle avoidance

In this scenario the vehicle navigated toward a goal a few hundred meters away in an urban environment. The navigation system successfully detected and avoided the raised edges of the road, thus producing a natural motion towards the goal. Additional simulations with cylindrical obstacles in the road were also carried out. Further information about the scenario is given in Figure 9 and its caption. Note that the odd stripping on the road is an artifact of the rendering of the output of the CityEngine tool.

The simulation ran at about half-real time; however, it should be noted that the simulation includes significant overhead for graphics, LIDAR modeling, path planning, navigation, etc., and that the multibody dynamics alone runs significantly faster than real time. Due to limited time, overall performance optimization was deferred in favor of developing more realistic urban and off-road ground scenarios, but we believe that real-time performance is well within reach.

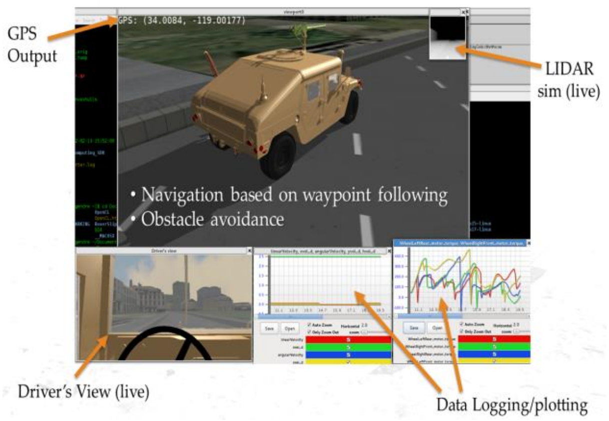

Figure 9: Scenario 1 - Urban driving with navigation and obstacle avoidance. From the top left of the figure going clock-wise, note these features: (a) GPS output, (b) the simulated LIDAR output is shown in gray levels, (c) data logging records and plots various vehicle telemetry, (d) a sample driver's view.

For robust operation with the navigation algorithm that was used for this scenario, the vehicle's speed was capped at 5m/s. This allowed the navigation algorithm to make more accurate estimates of the vehicle's trajectory and reduce the dynamic effects of the suspensions.

Scenario 2 - Urban driving - Lane Change Maneuver

In the second scenario, the vehicle performed a double lane-change maneuver in an urban environment.

The HMMWV was accelerated to about 20 m/s before making a lane change maneuver at 30 s, maintaining the lane for 20 s, and then a maneuvering back to its original lane (at 50 s). Figure 10 shows the vehicle during the lane change maneuver, and Figures

11-14 shows the vehicle speed, roll, pitch, yaw, lateral acceleration, and yaw rate during the maneuver.

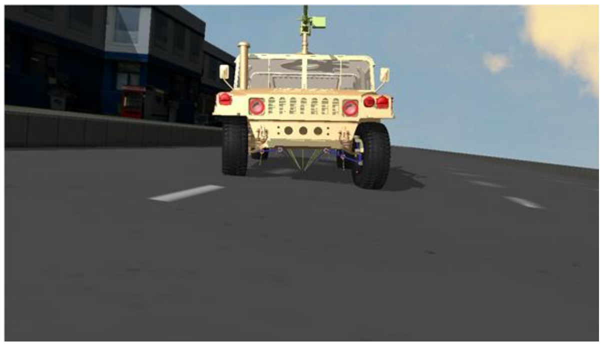
Figure 10: Scenario 2 - Lane change maneuver: Note the roll of the vehicle as it changes lanes

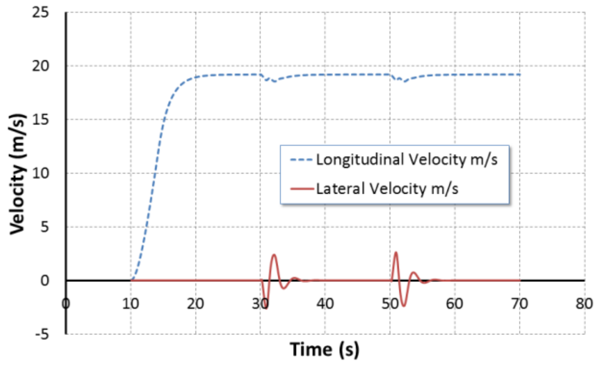

Figure 11: Scenario 2 – Vehicle speed during lane change maneuver

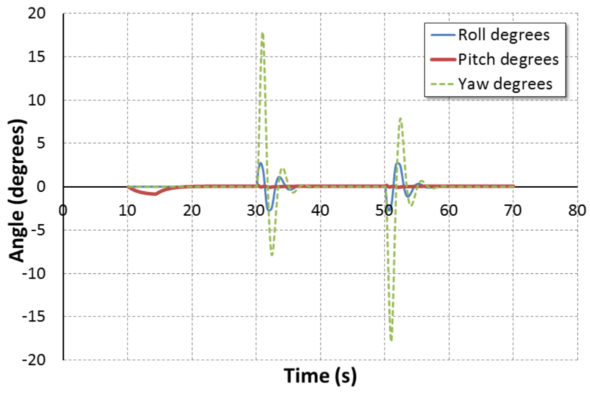

Figure 12: Scenario 2 – Roll, pitch, and yaw during lane change maneuver

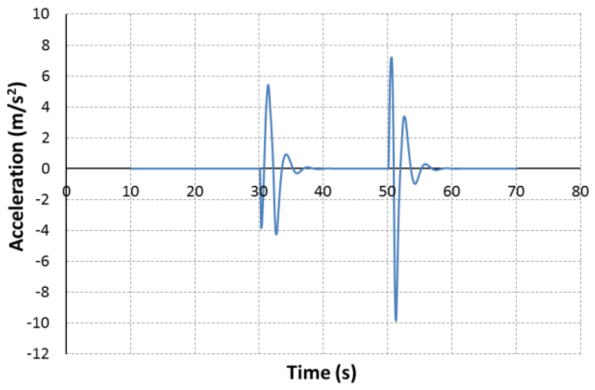

Figure 13: Scenario 2 – Lateral acceleration during lane change maneuver

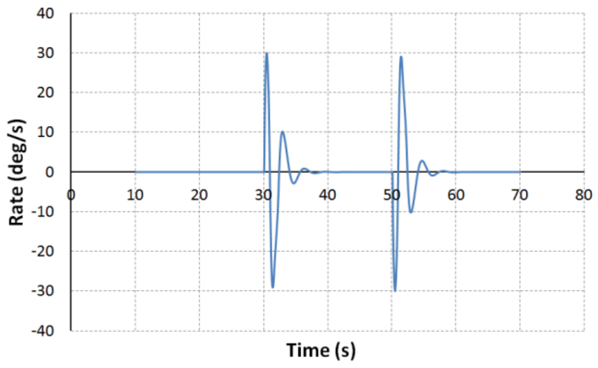

Figure 14: Scenario 2 – Yaw rate during lane change maneuver

Scenario 3 - Off-Road Driving Teleoperation

In Scenario 3, the vehicle was driven at various speeds on off-road terrain with a few trees to demonstrate vehicle behavior. The tire-terrain interaction is based on Bekker/Terazaghi models, which treats the tire as rigid and lumps all the tire-soil spring-damper interaction into the soil (as if the vehicle as driving over a soft or rubbery terrain). The terrain characteristics were chosen to represent relatively firm soil. In this scenario the vehicle was controlled using a joystick, without any autonomous navigation and obstacle avoidance algorithms.

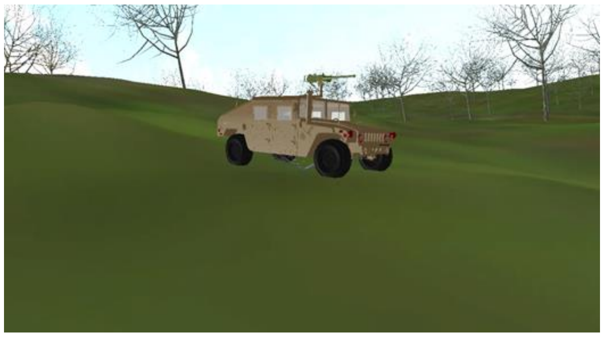

Figure 15: Scenario 3 - Off-road vehicle driving using joystick control

CONCLUSIONS

Current ground vehicle simulations used by the U.S. Army require a significant trade-off: If the vehicle model is of high fidelity, it operates too slowly to be useful for typical analysis purpose such as sensor testing and development. Only vehicles with very low modeling fidelity operate fast enough to be useful for typical autonomous vehicle analysis tasks.

The ROAMS HMMWV simulation successfully demonstrates that high fidelity multibody dynamics, terrain models, sensors, actuators, control, and navigation in urban and off-road scenarios can be modeled and run at speeds that are useful for vehicle analysis and design purposes. Even without significant attempts to optimize performance, the most complex ROAMS HMMWV simulation runs at near real-time. Although it was not demonstrated explicitly in the scenarios, the architecture of ROAMS allows models of varying complexity to be swapped in and out as desired for hardware, sensors, and control and navigation algorithms. All component models allow adjustment of run-time parameters to vary system.

Future Work

Although the current model captures many essential aspects necessary for accurate simulation of vehicle behavior, several potential improvements are topics for future work. Among these are the addition of an anti-roll bar and compliant rubber bushings in the suspension, better modeling of the drivetrain and steering column, and implementation of a tire model that has been verified for HMMWV tires. Further validation against measured vehicle data is also needed, to ensure that the model captures the true driving characteristics of the HMMWV.

Improvement of the sensor models and integration with more advanced closed-loop control and autonomy algorithms are further topics for future work.

ACKNOWLEDGEMENTS

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

We appreciate the support and encouragement of our U.S. Army TARDEC sponsors for making this work possible.

We thank Prof. Dan Negrut of the University of Wisconsin for providing the ADAMS HMMWV vehicle model mentioned in this paper.

We also wish to thank Brian Trease of JPL for assistance with the ADAMS HMMWV model.

Disclaimer

REFERENCES

[1] P. Jayakumar, W. Smith, B. A. Ross, R. Jategoankar and K. Konarzewski, "Development of High Fidelity Mobility Simulation of an Autonomous Vehicle in an Off-Road Scenario Using Integrated Sensor, Controller, and Multi-Body Dynamics," in 2011 NDIA Ground Vehicle Systems Engineering and Technology Symposium, Dearborn, Michigan, 2011.

[2] A. Jain, J. Guineau, C. Lim, W. Lincoln, M. Pomerantz, G. Sohl, R. Steele, "Roams: Planetary Surface Rover Simulation Environment", International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2003), (Nara, Japan), May 19-23, 2003.

[3] G. Sohl and A Jain, "Wheel-Terrain Contact Modeling in the Roams Planetary Rover Simulation," Fifth ASME International Conference on Multibody Systems, Nonlinear Dynamics and Control, Long Beach, CA, September 2005.

[4] A Jain, J. Cameron, C. Lim, J. Guineau, "SimScape Terrain Modeling Toolkit," Second International Conference on Space Mission Challenges for Information Technology (SMC-IT 2006), Pasadena, CA, July 2006.

[5] C. Lim, A. Jain, "Dshell++: A Component Based, Reusable Space System Simulation Framework", Third IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT 2009), Pasadena, CA, July 19-23, 2009.

[6] A. Jain, C. Crean, C. Kuo and M. Quadrelli, "Efficient Constraint Modeling for Closed-Chain Dynamics," in Second Joint International Conference on Multibody System Dynamics, Stuttgart, Germany, May 2012.

[7] J. Madsen, A Stochastic Framework for Ground Vehicle Simulation, Master's Thesis, University of Wisconsin - Madison, 2009.

[8] "CityEngine," Online: http://www.esri.com/software/cityengine.

[9] "Height Map Editor," Online: http://hme.sourceforge.net.

[10] "SPICE," Online: http://naif.jpl.nasa.gov/naif.

APPENDIX – MULTIBODY DYNAMICS WITH LOOP CONSTRAINTS

The following appendix includes an overview of the mathematical theory of the constraint embedding approach described in this paper and used in our HMMWV suspension model. More details about the following material can be found in [6].

As we have seen in the section on "Closed-Chain Suspension Modeling", each of the FA, TA and CE models consist of a tree topology sub-system subject to a set of closure constraints. The specific details of the decomposition vary with the model type. For the FA model, the tree system is simply a collection of independent bodies, while the set of closure constraints is large. The CE model represents the other extreme, where the entire model is a tree topology system, and there are no additional closure constraints. The TA model is a hybrid, with a maximal spanning tree based tree topology system, together with a minimal set of closure constraints. In this section we study the general approach to solving the closed-chain equations of motion for such models consisting of tree topology systems with closure constraints. Using N to denote the number of degrees of freedom for the tree sub-system, the minimal coordinates equations of motion for the tree-topology sub-system can be expressed as

$$\mathcal{M}(\theta)\,\ddot{\theta} + \mathcal{C}(\theta,\,\dot{\theta}) = \mathcal{T} \tag{1}$$

where the configuration dependent, symmetric matrix $\mathcal{M}(\theta) \in \mathcal{R}^{\mathcal{N}\times\mathcal{N}}$ is the *mass matrix* of the system, $\mathcal{C}(\theta,\,\dot{\theta}) \in \mathcal{R}^{\mathcal{N}}$ denotes the velocity dependent Coriolis and gyroscopic forces vector, and $\mathcal{T} \in \mathcal{R}^{\mathcal{N}}$ denotes the applied generalized forces. The mass matrix is positive-definite and invertible for tree-topology systems. Let n_c denote the dimensionality of the closure constraints on the system, Then there exists a $G_c(\theta, t) \in \mathcal{R}^{n_c\times\mathcal{N}}$ matrix and a $\mathfrak{U}(t) \in \mathcal{R}^{n_c}$ vector that defines the velocity domain constraint equation for the holonomic and non-holonomic closure constraints on the system as follows:

$$G_c(\theta, t)\,\dot{\theta} = \mathfrak{U}(t) \tag{2}$$

This differential form of the constraints is also referred to as a *Pfaffian* form. We assume that $G_c(\theta, t)$ is a full-rank matrix. Observe that Eq. 2 is linear in the $\dot{\theta}$ generalized velocity coordinates. These constraints effectively reduce the independent generalized velocities for the system from $\mathcal{N}$ to an ($\mathcal{N} - n_c$) dimensional linear space, The dynamics of closed-chain systems can be obtained by modifying the tree system dynamics in Eq. 1 to include the effect of the closure constraints via *Lagrange multipliers*, $\lambda \in \mathcal{R}^{n_c}$, as follows

$$\begin{aligned} \mathcal{M}(\theta)\,\ddot{\theta} + \mathcal{C}(\theta,\,\dot{\theta}) - G_c^*(\theta,t)\lambda &= \mathcal{T} \\ G_c(\theta,t)\,\dot{\theta} &= \mathfrak{U}(t) \end{aligned} \quad (3)$$

The $-G_c^*(\theta,t)\lambda$ term in the first equation represents the internal generalized constraint forces from the closure constraints[2].

By differentiating the constraint equation, Eq. 3 can be rearranged into the following descriptor form:

$$\begin{pmatrix} \mathcal{M} & G_c^* \\ G_c & \mathbf{0} \end{pmatrix} \begin{bmatrix} \ddot{\theta} \\ -\lambda \end{bmatrix} = \begin{bmatrix} \mathcal{T} - \mathcal{C} \\ \acute{\mathfrak{U}} \end{bmatrix} \quad (4)$$

$$\text{where} \quad \acute{\mathfrak{U}} \triangleq \dot{\mathfrak{U}}(t) - \dot{G}_c\,\dot{\theta} \in \mathcal{R}^{n_c}$$

One approach to solving the closed-chain dynamics equations of motion is to assemble the matrix on the left and the vector on the right in Eq. 4 and solve the linear matrix equation for the $\ddot{\theta}$ generalized accelerations. This is especially attractive for the FA model, since the $\mathcal{M}$ matrix for this case is block diagonal and constant. Indeed, the whole matrix is highly sparse for this case. This approach is analyzed in detail in reference. We on the other hand pursue an alternative Schur complement-based solution approach for the FA and TA models as described in the following lemma. For details about this, please see [6].

The closed-chain dynamics generalized accelerations in Eq. 4 can be expressed as

$$\ddot{\theta} = \ddot{\theta}_f + \ddot{\theta}_\delta \quad (5)$$

where the free generalized accelerations, $\ddot{\theta}_f$, the correction generalized accelerations, $\ddot{\theta}_\delta$, and the Lagrange multipliers, λ, are given by

$$\begin{aligned} \ddot{\theta}_f &\triangleq \mathcal{M}^{-1}(\mathcal{T} - \mathcal{C}) \\ \lambda &= -\left[G_c \mathcal{M}^{-1} G_c^*\right]^{-1} \gamma \\ \ddot{\theta}_\delta &\triangleq \mathcal{M}^{-1} G_c^*\,\lambda \end{aligned} \quad (6a-6c)$$

$$\textit{where} \quad \gamma \triangleq G_c\,\ddot{\theta}_f - \acute{\mathfrak{U}}$$

The $\ddot{\theta}_f = \mathcal{M}^{-1}(\mathcal{T} - \mathcal{C})$ term represents the generalized accelerations solution for the dynamics of the tree system while ignoring the closure constraints and, is therefore referred to as the free generalized accelerations. The term γ represents the acceleration-level constraint violation resulting from just the free dynamics of the system. The $G_c\mathcal{M}^{-1}G_c^*$ matrix in Eq. 6b is the Schur complement of the matrix on the left hand side of Eq. 4. An intuitive interpretation of Eq. 6b is that the constraint error spatial accelerations from the free-dynamics solution, together with the Schur complement matrix allow the computation of the constraint forces necessary to nullify the errors. Once the constraint forces are available, Eq. 6c uses them to obtain the generalized accelerations to correct the free system dynamics solution. In summary, the solution to the closed-chain forward dynamics thus conceptually involves the following steps:

1. Solve Eq. 6a for the $\ddot{\theta}_f$ free generalized accelerations.
2. Use $\ddot{\theta}_f$ and the $G_c\mathcal{M}^{-1}G_c^*$ Schur complement to solve for the Lagrange multipliers via Eq. 6b.
3. Use λ to solve Eq. 6c for the $\ddot{\theta}_\delta$ correction accelerations,
4. Compute the $\ddot{\theta}$ generalized accelerations using Eq. 5.

Only the first step is needed when closure constraints are absent. One numerical consequence of the use of non-minimal coordinates and closure constraints is that differential-algebraic equation (DAE) integrators, instead of the ODE integrators for tree systems, are required for the numerical integration of the accelerations and velocities. Furthermore, error control techniques are needed to manage the growth of constraint errors that can cause the system state to drift off of the constraint manifold.

[2] *Throughout this appendix, an asterisk superscript means transpose.*