Vectorized single-track model in Modelica for articulated vehicles with arbitrary number of units and axles

Peter Sundström[1], Bengt Jacobson[2], and Leo Laine[3]

[1]Modelon AB
[2]Chalmers University of Technology
[3]Volvo Group Trucks Technology

Abstract

A linear single-track model for articulated vehicles has been implemented. The model can represent an articulated vehicle with an arbitrary number of units each with an arbitrary number of axles. Lateral and yaw dynamics are in focus but longitudinal effects in couplings are also included. In the model, tire forces are linear with respect to slip angle. The couplings between units are represented as non-linear kinematic constraints which are valid for small and large articulation angles.

Four use cases are presented: Inverse dynamics for feedforward control, frequency responses when varying parameters, steady-state evaluations and dynamic simulation. For these use cases, four parametrizations of the model are used corresponding to a tractor with a semitrailer a truck with a dolly and a semitrailer, an A-double (tractor+semitrailer+dolly+semitrailer) and an approximate version of an airport baggage carrier with five full trailers.

Keywords: vehicle dynamics, vehicle models, articulated vehicles

1 Introduction

Simple vehicle dynamics models are very useful for basic analysis, rough parameter tuning and concept studies. By keeping models simple, one gains not only simulation speed but the possibility to for example invert the model for feedforward control. A simple model is however not always simple to model.

Several modeling approaches have been used to model articulated vehicles [1], [2], [3], [4]. For simple models, it is common to simplify the coupling equations using small angle approximations. This greatly simplifies the modeling effort but also invalidates the model for low-speed cases when articulation angles typically exceed the validity range of such an approximation. For multiple units, keeping the non-linear coupling equations usually require some kind of symbolic solver, such as Maple, Mathematica or Matlab's Symbolic toolbox. It is also common that the modeling is done for a specific vehicle combination with little room for extending to more units without reformulating the model.

Creating the model using Modelica one does not have to explicitly solve all equations and thus the model formulation can be kept simple while letting the Modelica tool take care of equation solving. Furthermore, by vectorizing the model with respect to the number of units and axles the model equations does not have to be expressed with a specific vehicle combination in mind, leaving that to the specific implementation.

2 Model description

A description of the model equations and structure is presented here. For brevity and readability of code snippets, `vector()` and `matrix()` function calls and some full library paths are left out. `MSL` is used when models are taken from the Modelica Standard Library, mainly from `Modelica.SIunits` and `Modelica.Blocks`.

2.1 Assumptions

As is normal for single-track models, the two tires on an axle are lumped as a single tire in the middle of the axle. Coupling points are assumed to not transfer torque between units. Slip angles are assumed to be small such that $\tan\alpha \approx \alpha$.

While the model accepts a variable input velocity,

typical longitudinal effects such as load transfer are not taken into account.

2.2 Parameter and variable structure

Parameters describing positions on the vehicle are all expressed relative to the first axle of the corresponding unit. This includes axle positions, center of gravity, c.g., positions and coupling point positions. Figure 1 shows the position parameters for a unit. Since states
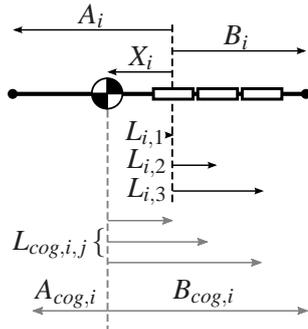
Figure 1: Geometric parameters of unit i. A_i and B_i are the distances from the first axle to the front and rear coupling points, respectively. X_i is the distance from the first axle to center of gravity. L_i, j is the distance from axle 1 to axle j on unit i.

are defined at the center of gravity of each unit, position parameters are recalculated relative to this, e.g. $A_{cog} = A - X$.

Parameters and quantities related to axles, such as tire forces and axle positions, are defined as matrices with dimensions $[nu, na]$ where nu is the number of units in the combination and na denotes the maximum number of axles on any of the units. If the number of axles on a unit is less than na the unused elements will be set to zero and are thus disregarded. Parameters and quantities related to units, such as motion states and masses, are defined as vectors of length nu.

2.3 Tire forces

Given the small angles assumption, the slip angle for a single tire is defined as

$$\alpha = \frac{v_y + L_{cog}\omega_z}{v_x} - \delta \quad (1)$$

where L_{cog} is the distance from the center of gravity to the axle that is being considered, v_y and ω_z are the lateral velocity and yaw rate of the unit in question and δ is the steering angle. In the model code, the slip angle matrix for all axles is defined as

```
alpha = ((vy*ones(1,na)
    +Lcog.*(wz*ones(1,na)))
    ./(vx*ones(1,na))-delta);
```

The lateral tire forces can then be calculated as

```
Fyw = -C.*alpha;
```

where `C` is the axle cornering stiffness matrix. Together with the longitudinal forces, `Fxw`, the tire forces can be transformed to the vehicle coordinate system with the steering angle `delta`

```
Fx = Fxw.*cos(delta)-Fyw.*sin(delta);
Fy = Fxw.*sin(delta)+Fyw.*cos(delta);
```

A variable `Fxd` is defined as the drive force needed at each driven axle to maintain the input velocity. This force is applied at each driven axle:

```
for i in 1:nu loop
   for j in 1:na loop
      if driven[i,j] then
         Fxw[i,j]=Fxd;
      else
         Fxw[i,j]=0;
      end if;
   end for;
end for;
```

This force is implicitly determined to satisfy the rest of the model equations.

2.4 Coupling constraints

The constraints in the couplings state that the global velocity vector of the rear coupling on the unit in front should be the same as that of the front coupling point on the unit behind. The Modelica language allows for a series of equations to be defined using for loops. Here, the constraint that the velocities at the front coupling of unit i should be the same as that of the rear coupling on unit i-1 is defined by looping over all the couplings:

```
for i in 1:nu-1 loop
vx[i+1] = vx[i]*cos(theta[i])
    -(vy[i]+Bcog[i]*wz[i])*sin(theta[i]);
vy[i+1]+Acog[i+1]*wz[i+1]=
    (vy[i]+Bcog[i]*wz[i])*cos(theta[i])
    +vx[i]*sin(theta[i]);
end for;
```

Equations for the coupling forces are not formulated explicitly, the velocity constraints in the coupling to-

gether with force and moment balances gives enough information for an implicit calculation. Figure 2 shows how the coupling cut forces are applied to the different units affected by the coupling. The forces
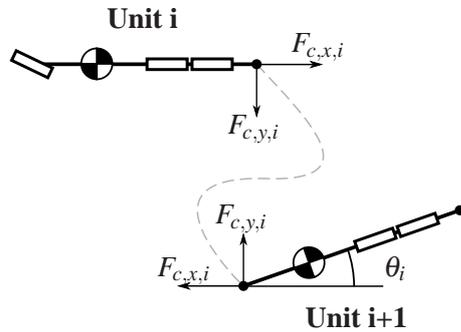
Figure 2: Example of cut forces in coupling between unit i (front) and i+1 (rear)

in each coupling are defined in the coordinate system of the front unit which it affects with negative sign. The rear unit in the coupling is affected by the forces with positive sign transformed by the articulation angle. So `Fcx[i]` pulls rearward on unit `i` and `Fcx[i]*`**`cos`**`(theta[i])` pulls forward on unit `i+1`. Similarly, `Fcy[i]` pulls in the negative y direction (rightwards) on unit `i` and `Fcy[i]*`**`cos`**`(theta[i])` pulls towards the left on unit `i+1`.

2.5 Steady state mode

To simplify steady-state analysis, substitute parameters are defined for each state derivative:

```
  MSL.Acceleration[nu] d_vx;
  MSL.Acceleration[nu] d_vy;
  MSL.AngularAcceleration[nu] d_wz;
  MSL.AngularVelocity[nu-1] d_theta;
```

If the model is to be run in steady-state, all derivatives are set to zero. For dynamic simulations they are set to the derivative of their corresponding state variable:

```
if steadystate then
   d_vx=zeros(nu);
   d_vy=zeros(nu);
   d_wz=zeros(nu);
   d_theta=zeros(nu-1);
else
   d_vx=der(vx);
   d_vy=der(vy);
   d_wz=der(wz);
   d_theta=der(theta);
end if;
```

2.6 State equations

By using matrices and vectors, the state equations can be formulated as matrix equations for the entire vehicle combination instead of writing separate equations for each unit. In the axle force matrices, a row, `i`, contains all the axle forces on unit `i`. To get the force sum on each unit, the force matrix is multiplied by a column vector of ones of length `na`:

$$\begin{bmatrix} F_{y,11} & \cdots & F_{y,1n_a} \\ \vdots & \ddots & \vdots \\ F_{y,n_u1} & \cdots & F_{y,n_un_a} \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} F_{y,11}+...+F_{y,1n_a} \\ \vdots \\ F_{y,n_u1}+...+F_{y,n_un_a} \end{bmatrix} \quad (2)$$

For coupling forces column vectors `[Fcx;0]` and `[Fcy;0]` are defined for the rear coupling forces, the 0 element meaning no rear coupling force on the rearmost unit. Similarly, vectors `[0,Fcx]` and `[0,Fcy]` are defined for the front coupling forces where 0 here means no front coupling force on the first unit. Then, the force balance equations for the whole combination can be written as a matrix equation using element-wise multiplication. The lateral and longitudinal force balances are written as:

```
ay=d_vy+vx.*wz;
m.*ay=Fy*ones(na,1)-[Fcy;0]
   +[0;Fcx].*sin([0;theta])
   +[0;Fcy].*cos([0;theta]);
```

```
ax=d_vx-vy.*wz;
m.*ax=Fx*ones(na,1)-[Fcx;0]
   +[0;Fcx].*cos([0;theta])
   -[0;Fcy].*sin([0;theta]);
```

The moment balance around the yaw axis is done in the same way with the added moment arms for the different forces:

```
I.*d_wz=Lcog.*Fy*ones(na,1)
   -Bcog.*[Fcy;0]
   +Acog.*([0;Fcx].*sin([0;theta])
   +[0;Fcy].*cos([0;theta]));
```

where `Acog` and `Bcog` are the distance from c.g. to the front and rear couplings, respectively and `Lcog` is the distance from c.g. to each axle.

2.7 Inputs

The inputs to the model are the steering angles at all axles and the velocity of the first unit.

```
input MSL.RealInput[nu,na] delta_in;
input MSL.RealInput vx_in;
```

Due to the kinematic constraints in the coupling, the derivative of the `vx_in` input will be needed. To allow velocity as input when exporting the model, the derivative of `vx_in` is set to zero with annotated derivatives using a QuasiStatic function from the Modelon library:

```
vx[1] =
    max(0.1,QuasiStatic.scalar(vx_in));
```

The output from the `QuasiStatic.scalar(vx_in)` function has the same value as `vx_in` but with zero derivative. Here, the velocity is also set to be minimum 0.1 to protect sideslip calculations from division by zero.

3   Use cases

Four use cases are presented as examples. For each use case a different vehicle combination is used to show how the model can be parametrized for different vehicles.

A tractor-semitrailer model is inverted for feedforward control, a truck-dolly-semitrailer is linearized and its frequency response is studied, a tractor-semitrailer-dolly-semitrailer is run in steady-state to evaluate its off-tracking characteristics and a five-trailer airport baggage carrier is run in a dynamic simulation. Figure 3 shows the four vehicles used.
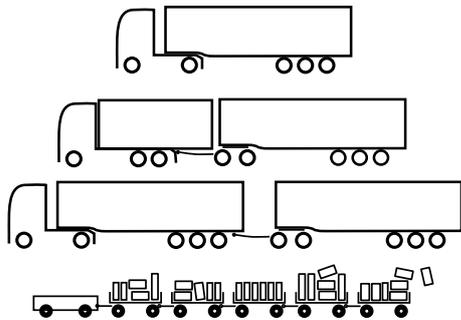
Figure 3: The four vehicle combinations used.

The Functional Mockup Interface, FMI, allows modeling and simulation to be performed in separate tools. Model development has been done in Dymola and all simulations and plotting are done in Matlab/Simulink using FMI Toolbox. The only parameters that need to be set before exporting models are the `steadystate` setting and the number of units, `nu`, and axles, `na`, as they change the structure of the model. Other parameters such as axle positions and cornering stiffnesses are free to set when using the exported model.

3.1   Inverse dynamics

By using the InverseConstraints block in the Modelica Standard Library, the inverse dynamics of the model can be solved for. To use the model for feedforward control, we select the lateral acceleration `a_y` of the first unit as input and the steering angle at the front axle `delta[1,1]` as output. Figure 4 shows the block diagram of the inverse model.
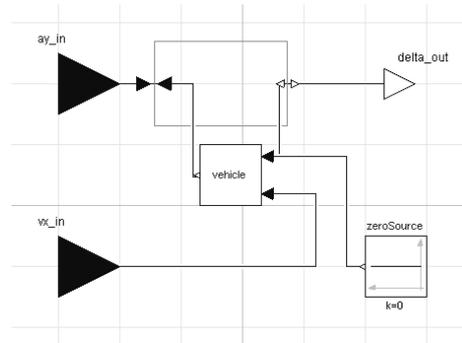
Figure 4: Block diagram of inverse model. Inputs to the model are lateral acceleration and longitudinal velocity at the first unit. Output is the required steering angle at the first axle.

To test the inverted model, a triangular acceleration signal of 0.2 Hz frequency and 2 m/s^2 amplitude is sent as input. The velocity is 50 km/h. The steering output from the inverted model is then input to a model with normal causality to verify that the lateral acceleration achieved will be correct. Figure 5 shows that the lateral acceleration output from the controlled model is indeed the same as the desired lateral acceleration input. The resulting steering command is shown in figure 6.
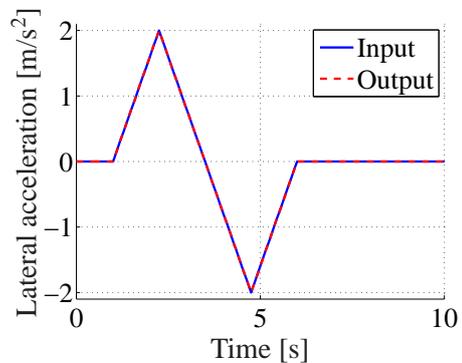
Figure 5: Desired lateral acceleration input and output from vehicle controlled with steering output from inverted model.
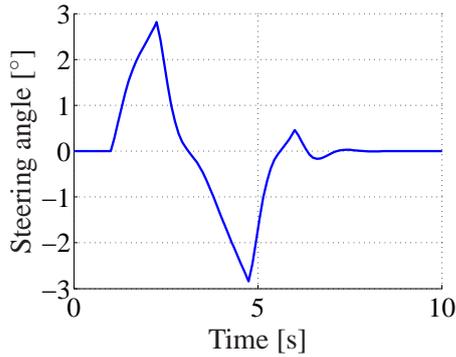
Figure 6: Steering angle from inverted model.

3.2 Frequency response

The model can easily be linearized to find the frequency response. An important use case for this is to find how different parameters affect the frequency response. Here, the coupling position between the tractor and the trailer is varied to show how this affects the stability of the vehicle combination.

The truck-dolly-semitrailer combination is linearized for straight forward driving at 80 km/h. Figure 7 shows the baseline gain from steering angle to the yaw rates of the different units in the truck-dolly-semitrailer combination.
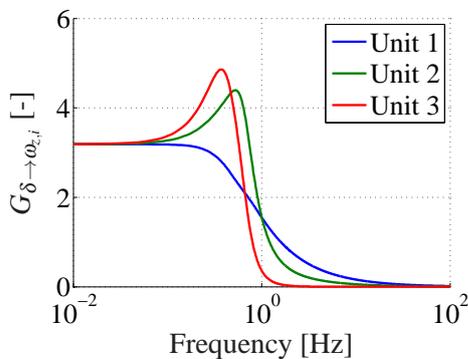
Figure 7: Gain from steering angle at the front axle to the yaw rates of the different units for the baseline parametrization of the truck-dolly-semitrailer combination.

Often the rearward amplification is an important measure of vehicle performance. This is defined as the gain from the first unit yaw rate to each of the towed units' yaw rates. Figure 8 and figure 9 show how the rearward amplification of the dolly and semitrailer changes as the coupling position on the truck is changed. The baseline position is 0.53 m behind the rear axle. As the coupling moves rearward the amplification increases.
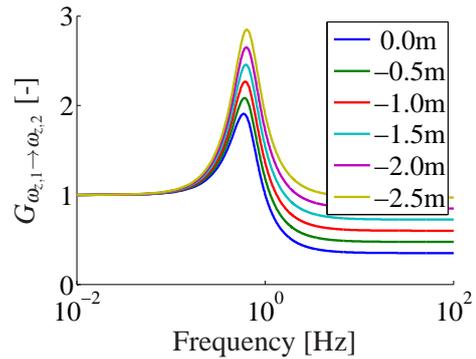
Figure 8: Yaw rate gain from first unit to second unit when moving coupling position on first unit. Coupling further rearward (negative adjustment) gives higher amplification of yaw rate.
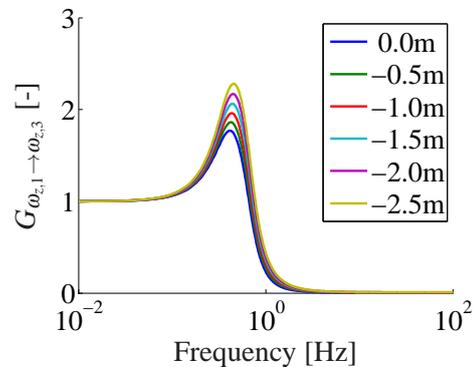
Figure 9: Yaw rate gain from first unit to third unit when moving coupling position on first unit. Coupling further rearward (negative adjustment) gives higher amplification of yaw rate.

3.3 Steady-state off-tracking

Off-tracking is an important property of a vehicle with trailers. It is usually defined as the difference in curve radius between the towing unit and the trailers. In the model, the instantaneous curvature is calculated for all axles as

```
for i in 1:nu loop
    for j in 1:na loop
        curvature[i,j] = wz[i]
        /sqrt((vy[i]+Lcog[i,j]*wz[i])^2
        +vx[i]^2);
    end for;
 end for;
```

To show how off-tracking varies with vehicle speed, the vehicle is driven at a constant curve radius of 100 m for velocities from 1 to 80 km/h. Figure 10 shows how the curve radius of the last axle of each trailer compares to that of the first axle on the tractor. The typical

characteristic is that trailers track cuts into the curve at low speed due to vehicle kinematics and geometry. At high speed the trailers track a larger curve radius due to the increased lateral load causing higher sideslip.
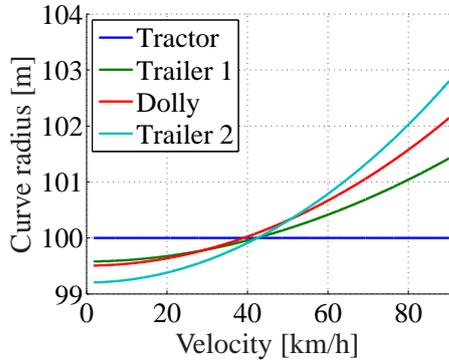
Figure 10: Curve radius of the first tractor axle and the last axle of each towed unit for varying velocity.

Off-tracking for a fixed vehicle speed also varies with the curve radius of the first unit. To study this, the vehicle is run at a fixed speed of 10 km/h and curve radius of the first unit varies from 15 to 100 m. Figure 11 shows how off-tracking varies with the curve radius of the first axle. At this relatively low speed, vehicle kinematics cause more inward off-tracking at smaller curve radii.
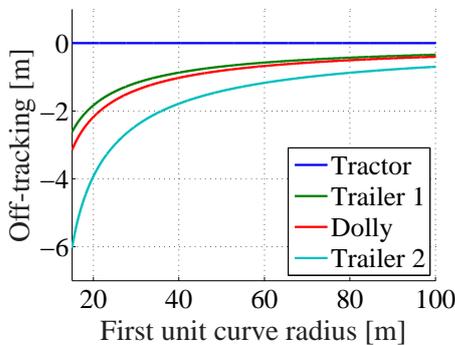
Figure 11: Off-tracking of the last axle of each towed unit in the A-double. First unit speed is fixed and curve radius varies. Positive off-tracking is defined outwards in the curve.

3.4 Dynamic simulation

The baggage carrier is simulated with a single period sine steering input of 5° amplitude and 0.3 Hz frequency at 18 km/h. Figure 12 shows the yaw rates of all the units in the airport baggage carrier and figure 13 shows the position of each axle.

The parameters are only approximated and are not measured or estimated from a specific real vehicle.
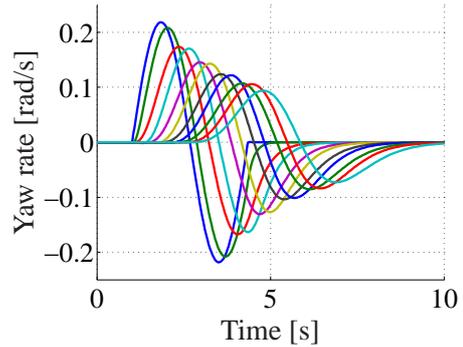
Figure 12: Yaw rates of all the units in the baggage carrier train
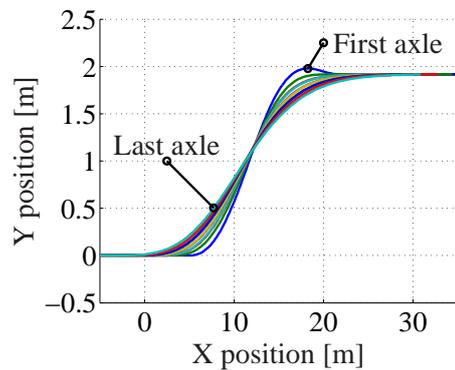
Figure 13: Positions of each axle in the baggage carrier train.)

The simulation results show that the model can represent multi-trailer vehicles without additional modeling effort.

4 Conclusion

This work shows that Modelica is useful for formulating simple vehicle models. Using Modelica, articulated vehicle combinations with arbitrary number of units and axles can be modeled with the same level of abstraction as when modeling a non-articulated single-track vehicle model. Tool independent export and import possibilities within the Functional Mockup Interface greatly simplifies the use of models like this in control design and concept studies.

By incorporating simple-but-relevant models together with higher fidelity models in development processes infeasible concepts can be ruled out early and only the most promising ones can be carried on to more detailed analyses.

References

[1] M. Levén, A. Sjöblom, M. Lidberg, and B. Schofield, "Derivation of linear single-track truck-dolly-semitrailer model with steerable axles," Department of Applied Mechanics, Chalmers University of Technology, Tech. Rep. 2011:09, 2011.

[2] M. Gäfvert and O. Lindgärde, "A 9-dof tractor-semitrailer dynamic handling model for advanced chassis control studies," Department of Automatic Control, Lund Institute of Technology, Tech. Rep. ISRN LUTFD2/TFRT–7597–SE, 2001.

[3] S. Kharrazi, "Steering based lateral performance control of long heavy vehicle combinations," Ph.D. dissertation, Chalmers University of Technology, 2012.

[4] D. Vazquez-Vega, "Directional analysis of an a-train double with damped articulation," Master's thesis, Concordia Unversity, Montreal, 2000.