I. Kossenko
Moscow State University of Service, Russia
Implementation of Unilateral Multibody Dynamics on Modelica
pp. 13-23

Implementation of Unilateral Multibody Dynamics on Modelica

Ivan I. Kossenko
Moscow State University of Service, Department of Engineering Mechanics
Glavnaya str., 99, Cherkizovo-1, Moscow reg., 141221, Russia

Abstract

The problems of computer modeling and simulation of dynamics for multibody systems consisting of rigid bodies with unilateral constraints (MBSUC) are considered in the scope of the obstacles to overcome ones related to the variation of structure for equations of motion. The approach to modeling the MBSUC dynamics based on Modelica language is described.

The approach allowing to avoid the growth of the model structural complexity is described. This approach actively uses the algorithmic features of Modelica and its Dymola compiler. On this way the large number of objects corresponding to different closed systems of DAEs (states of hybrid automata) is replaced by only one object. For this object constraint components vary their states dynamically during the simulation process.

Another problem of the similar level of complexity relates to the accuracy of simulation is solved here with the set of special regularization procedures. These procedures concern particularly transitions of the unilateral constraint: from disconnected state to contact, from rolling to slipping.

Other methods to improve the quality of the MBSUC dynamics simulation are also under consideration.

Keywords: unilateral dynamics; multibody systems; simulation; dry friction; impacts; regularization; acausal modeling

1 Introduction

Mechanical system subjected to unilateral constraints exhibits behavior considerably more complicated than the system subjected to the bilateral ones. One can find in such a case new dynamical properties connected with irregular character of appropriate systems of DAEs. Let us develop the approach proposed in [1]. There the Modelica library of classes oriented to simulation the sparse dynamics of multibody systems has been developed. We can consider now this library as a set of the new generation models allowing description of unilateral constraints.

Let us suppose that some of constraints are unilateral. For definiteness and simplicity we state the following assumptions: (a) unilateral constraint is implemented as a contact of outer surfaces bounding two rigid bodies; (b) surfaces supposed being regular i. e. the normal vector is always properly defined; (c) the contacting surfaces interact within the model of Coulomb friction for continuous motions as well as for impacts.

For simplicity we investigate the MBSUC comprising only two bodies, *A*, and *B*. Moreover, we suppose that the body *A* is a fixed horizontal surface, and the heavy convex body *B* is bounded by ellipsoidal surface. These assumptions are not obstacles for generality of the developed MBSUC models.

2 Basic Ideas

According to the approach applied in [1] let us represent the constraint as an object providing information communications between the objects of bodies *A* and *B*. Such communications are implemented indirectly using the kinematic and wrench connectors. Information communications are “filtrated” through the mechanism of constraint equations encapsulated in the object *C*, see Figure 2.1.

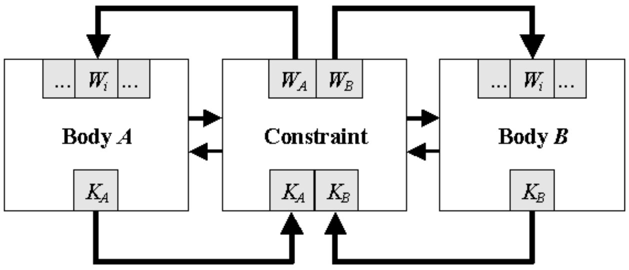

Figure 2.1: Architecture of Unilateral Constraint

Besides the bidirected connections applied in [1] let us add to the model the set of directed connections. Assume that these connections are able to transmit

the impact signals arising in objects of unilateral constraints all over the MBSUC, namely throughout its connected components. These signals play role of strobing ones for recalculation of velocities in the MBSUC.
The nature of unilateral constraint allows us to describe it with the fundamental state variable. This variable takes one of three values: "Flight", "Sliding", "Rolling" at any time instant. The sense of the enumerated values is transparent. The state "Flight" means that the constraint is not stretched at the considered instant, i. e. the bodies aren't in touch and freely fly one relative to another. As state variable has one of values "Sliding" or "Rolling" then bodies supposed to be in a contact. The difference is that the first state permits the relative slipping of the bodies but the second one doesn't.

Example 2.1 *Consider the set of n balls in a billiard pool. The system comprises $n+1$ rigid bodies: n balls and the surface of the pool table. Vertical surfaces around the table are neglected for simplicity. All bodies enumerated can encounter mutually, slip, and roll. The correct description of this MBSUC involves the specification of $m = n(n+1)/2$ unilateral constraints. Since each constraint can be in one of three states, then the whole MBSUC comprises $3^m = 3^{\frac{n(n+1)}{2}}$ states. For the pool with three balls we obtain the total value of $3^6 = 729$ states.*

2.1 Constraint Geometry

Let us use here the same as in [1] the dynamics of a rigid body translational–rotational motion. However the representation of mechanical constraint model undergoes here essential changes. We use the so called complementarity rules [2] as a base for the unified description of the unilateral constraint. By virtue of complementarity rules any constraint is always defined by the three scalar equations. In order to derive these equations let us consider the local geometry of the problem, see Figure 2.2.
The base body of MBSUC supposed to be connected with the absolute frame $O_0x_0y_0z_0$ (AF) fixed in the inertial space, $O_\alpha x_\alpha y_\alpha z_\alpha$ is the frame BF_α fixed in the body $\alpha \in \{A,B\}$. The outer surfaces Σ_α are defined by the equations

$$f_\alpha(\mathbf{r}_\alpha) = 0 \quad (\alpha = A, B).$$

with respect to appropriate BF_α whose axes are coincident to the principal central axes of inertia. In AF

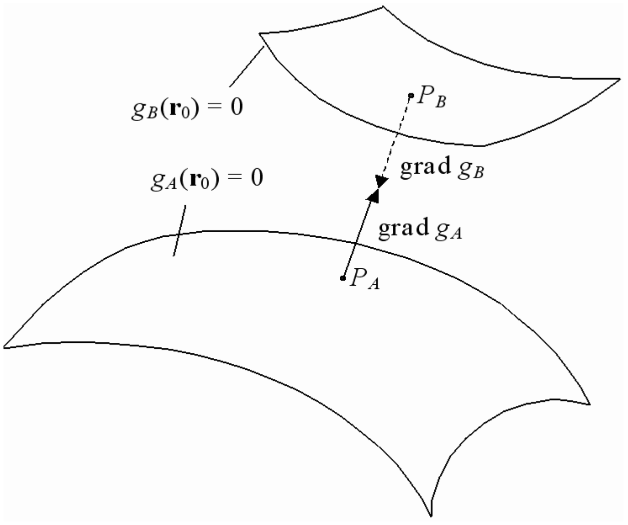

Figure 2.2: Area of Constraint

these the equations read

$$g_\alpha(\mathbf{r}_0) = f_\alpha[T_\alpha^*(\mathbf{r}_0 - \mathbf{r}_{O_\alpha})] = 0 \quad (\alpha = A, B).$$

Here $\mathbf{r}_{O_A} = O_0O_A$, $\mathbf{r}_{O_B} = O_0O_B$, T_A, T_B are the orthogonal matrices determining orientation of the BF_A and BF_B with respect to the AF. An asterisk denotes the matrix transposition. The functions $g_A(\mathbf{r}_0)$, $g_B(\mathbf{r}_0)$ depend upon the time indirectly through the variables $\mathbf{r}_A$, $\mathbf{r}_B$, T_A, T_B.
The constraint object of our model has to compute at each current instant the points $P_A \in A$ and $P_B \in B$ realizing the minimal distance between the bodies. These points depend on relative orientation of the bodies. By virtue of above assumptions such points are to be evaluated in a unique way. Denote by $\mathbf{r}_{P_A}$, $\mathbf{r}_{P_B}$ the radii vectors of these points with respect to AF. The simple geometric reasons imply the following system of algebraic equations

$$\begin{aligned} \operatorname{grad} g_A(\mathbf{r}_{P_A}) &= \lambda \cdot \operatorname{grad} g_B(\mathbf{r}_{P_B}), \\ \mathbf{r}_{P_A} - \mathbf{r}_{P_B} &= \mu \cdot \operatorname{grad} g_B(\mathbf{r}_{P_B}), \\ g_A(\mathbf{r}_{P_A}) &= 0, \\ g_B(\mathbf{r}_{P_B}) &= 0. \end{aligned} \tag{2.1}$$

The gradients of the functions g_A and g_B read

$$\operatorname{grad} g_\alpha(\mathbf{r}_{P_\alpha}) = T_\alpha \operatorname{grad} f_\alpha[T_\alpha^*(\mathbf{r}_{P_\alpha} - \mathbf{r}_{O_\alpha})], \tag{2.2}$$

where $\alpha = A, B$. The system (2.1) consists of eight scalar equations with respect to eight scalar variables: $x_{P_A}, y_{P_A}, z_{P_A}, x_{P_B}, y_{P_B}, z_{P_B}, \lambda, \mu$, where λ, μ are auxiliary variables. The equations (2.1) are in use either without or with a presence of the contact of bodies A, B. In the latter case the equation $\mu = 0$ is in use instead of one of the surfaces equations.

According to computational experience it is more reliable and convenient to use the equations of constraints (2.1) in a differential form. Such an approach is used frequently also for analyzing of properties of mechanical systems.
Normal vector

$$\mathbf{n}_A = \operatorname{grad} g_A \big/ |\operatorname{grad} g_A| \qquad (2.3)$$

will play an important role in the further course. Normal for an outer surface of the body A is chosen here for definiteness. One can use the vector $\mathbf{n}_B$ as well.

2.2 Complementarity Rules

Let us perform a unified description of the unilateral constraint using kinematic and/or force equations. Denote by $\mathbf{F}_A$ the force acting on the body A from the body B. And by $\mathbf{F}_B$ denote the force acting on the body B from one of A vice versa. Each force cited acts at the point P_α, $\alpha = A, B$. In addition, let us introduce auxiliary notations

$$F_{An} = (\mathbf{F}_A, \mathbf{n}_A), \quad \mathbf{F}_{A\tau} = \mathbf{F}_A - F_{An}\mathbf{n}_A,$$

$$\mathbf{v}_r = \mathbf{v}_{P_A} - \mathbf{v}_{P_B}, \quad v_{rn} = (\mathbf{v}_r, \mathbf{n}_A), \quad \mathbf{v}_{r\tau} = \mathbf{v}_r - v_{rn}\mathbf{n}_A. \qquad (2.4)$$

If the bodies are not in touch and the constraint is in the state “Flight” then the force of reaction is equal to zero. Thus we have three scalar equations. To unify the system of constraint equations and to take into account arbitrary directions of the normal $\mathbf{n}_A$ let us introduce auxiliary scalar variable κ such that

$$F_{An} = 0, \quad \mathbf{F}_{A\tau} - \kappa\mathbf{n}_A = \mathbf{0}.$$

Then the system of four equation with four unknown variables F_{Ax}, F_{Ay}, F_{Az}, κ is obtained.
If the bodies are in touch then the condition $F_{An} = 0$ is substituted by the kinematic one $v_{An} = 0$. States “Sliding” and “Rolling” differ from each other by conditions in a tangent plane. Implementation of the Coulomb friction model is supposed for the simplicity. Then the equation of the force balance in the tangent space reads

$$\mathbf{F}_{A\tau} - d \cdot F_{An}\mathbf{v}_{r\tau} \big/ |\mathbf{v}_{r\tau}| - \kappa\mathbf{n}_A = \mathbf{0}, \qquad (2.5)$$

where d is the coeffitient of friction.
For rolling the tangent velocity is:

$$\mathbf{v}_{A\tau} - \kappa\mathbf{n}_A = \mathbf{0}.$$

2.3 Regularization of the Coulomb Friction

In the case of sliding the model equation (2.5) “works” properly if the relative velocity isn’t very small. However the problem of regularization for the equation of constraint (2.5) arises at the instance of transition from “Rolling” to “Sliding”. It turns out that one can apply here the known approximation for Coulomb’s friction using regularized expression for the tangent force

$$\mathbf{F}_{A\tau} - \kappa\mathbf{n}_A = d \begin{cases} F_{An}\mathbf{v}_{r\tau} \big/ |\mathbf{v}_{r\tau}| & \text{as} \quad |\mathbf{v}_{r\tau}| > \delta, \\ F_{An}\mathbf{v}_{r\tau}/\delta & \text{as} \quad |\mathbf{v}_{r\tau}| \le \delta, \end{cases}$$

where one supposes that $\delta \ll 1$.
It is known [3] that in this case the solution of the regularized problem remains close to the solution of the original one at the asymptotically large time intervals. Implementation and further simulation show that this closeness holds with the very high degree of accuracy. Such an approach resolves completely the problem of modeling for accurate transitions between states of “Sliding” and “Rolling”.

2.4 Simulation of Impacts

Let us suppose that the unilateral constraint is allowed to undergo an impact in any possible states. In state “Flight” the impact arises at the instant of bodies contact if normal component of the relative velocity v_{rn} for encountering points is not very close to zero. It is the case of the so called direct impact. However in MBSUC consisting of several bodies impact pulses can propagate through the connected components of the system and force it to disconnect of any constraints. This leads to the switch of the whole MBSUC to an another state. Such a case we can consider as an indirect impact.
The constraint model proposed allows the possibility both direct and indirect impacts. Let us consider the equations of the impact theory encapsulated in the objects of the constraint structure, see Figure 2.1. All these algebraic equations are carried out for all the time of simulation. From time to time impact events arising inside the differential part of the whole model strobe “reading” of impact increments for the velocities from the impact algebraic subsystem and instanteneous change of velocities inside the dynamical subsystem.
Thus the equations

$$m\Delta\mathbf{v} = \mathbf{S}, \quad I\Delta\boldsymbol{\omega} = \mathbf{T}, \qquad (2.6)$$

are encapsulated in objects A and B of the “Rigid Body” class. Here $\Delta\mathbf{v}$, $\Delta\boldsymbol{\omega}$ are the increments of the center of mass velocity and angular velocity of the body, **S**, **T** are correspondingly the total impulse and angular impulse acting on the rigid body belonging to MBSUC. Note that the first equation of the system (2.6) is written in AF. The second one is written, as usually, in appropriate BF_α.

Constraint object, C in Figure 2.1, encapsulates the simplest impact model with dry friction and the Newtonian model for the normal impact

$$\begin{array}{rcl}
\Delta\mathbf{v}_{P_\alpha} & = & \Delta\mathbf{v}_{O_\alpha} + [\Delta\boldsymbol{\omega}_\alpha, \mathbf{r}_{P_\alpha} - \mathbf{r}_{O_\alpha}], \\
\Delta v_{P_\alpha n} & = & (\Delta\mathbf{v}_{P_\alpha}, \mathbf{n}_A), \\
\Delta v_{rn} & = & -(1+k)v_{rn}, \\
\Delta v_{P_B n} & = & \Delta v_{P_A n} - \Delta v_{rn}, \\
\Delta\mathbf{v}_{P_\alpha \tau} & = & \Delta\mathbf{v}_{P_\alpha} - \Delta v_{P_\alpha n}\mathbf{n}_A, \\
S_{An} & = & (\mathbf{S}_A, \mathbf{n}_A), \\
\mathbf{S}_{A\tau} & = & \mathbf{S}_A - S_{An}\mathbf{n}_A,
\end{array} \tag{2.7}$$

where the restitution coefficient is equal to k.

To make the model of impact with friction more realistic we apply the simplified formula for the impact impulse. It is similar to the regularized formular for the tangent force in the case of slipping with dry friction. Let us note that there exist more realistic models of impact with the Coulomb friction [4] (see [5] for comprehensive survey). However they are much more complicated. These models are suited for the single impact of two bodies only. But we are interested in a general case of MBSUC consisting of several bodies not only of two ones.

2.5 Regularization of Transition between the States of Flight and of the Contact

The most important property of the model developed consists of the possibility of exact calculation of impact instants and the instants of the change of state. This property plays a crucial role for the quality of the model. The landing on the constraint is possible in particular if restitution coefficient satisfies the condition $k < 1$. In this case time intervals between impacts tend to zero as well as the amplitudes of jumps after successive impacts. Thus for exact determination of the landing instant there exists a technological restriction: limit of smallness for the value of the integrator time step.

The change of independent variable, which regularizes the time, gives the resolution of the problem. Indeed, let us consider approximate model of dynamics in a vicinity of the landing instant. In this case we can restrict ourselves to analysis of the relative motion for points P_A and P_B in normal direction. Let us assume that the normal relative acceleration $a_{rn} = dv_{rn}/dt$ approximately is a constant. Then the relative normal motion of the points A and B is similar to the bouncing ball in field of constant acceleration a_{rn} in the vicinity of the landing instant.

Thus the height of jumps obeys the known formula $h = 0.5v_{rn}^2/a_{rn}$. Hence the instant of transition to contact is defined by the condition when h becomes less then the given value of the tolerance for the constraint feasibility.

The time between two impacts can be also approximately computed with the known formula $T = 2\left|v_{rn}/a_{rn}\right|$. This value tends to zero with each new impact leading to the loss of an accuracy of simulation.

Way out of a situation is the transfer to new independent variable τ such that the duration between successive impacts would stay of order one. The simplest solution of this problem is the map $t \mapsto \tau$ according to the scalar differential equation $dt/d\tau = T$. Such an approach is found to be sufficiently reliable. Moreover it is easy to control the accuracy of the model.

3 Implementation

When constructing the model of MBSUC the main task is to develop the Modelica code allowing to switch different constraint states inside the same object, see Figure 2.1. It was found the problem can be resolved using so called acausal [6] approach to build the system of DAEs for the resulting model. Alternatively if one uses the causal appoach then the structural complexity of a model code can increase avalanchely. To make sure of this it is sufficient to remind our example about three balls in a billiard pool. If each state of the mechanical system corresponding to the closed system of DAEs is instantiated as an object inside the container of the hybrid automata model then very soon developer will encounter with the problem of large complexity even for a number of balls small enough.

Conversely within the acausal approach there exists a possibility to construct the model of MBSUC at the same complexity level as for mechanical system subjected to bilateral constraints only. In this case all variety of MBSUC states is provided by internal capabilities of the constraint objects and, as usually it is implemented with help of an analytical precompiler.

3.1 Connectors

To connect objects we use the classes of kinematic and wrench ports as before [1]. In addition, new connectors are able to transport data of the velocities increments and the impact impulses. Codes of the corresponding derived classes read

```
connector KinematicPortImpacts
  extends KinematicPort;
  SI.Velocity Deltav[3];
  SI.AngularVelocity Deltaomega[3];
end KinematicPortImpacts;

connector WrenchPortImpacts
  extends WrenchPort;
  SI.Impulse ImpactForce[3];
  SI.AngularImpulse ImpactTorque[3];
end WrenchPortImpacts;
```

To transmit impact signals throughout the MBSUC one uses standard signal input and output ports:

```
  Interfaces.BooleanInPort,
  Interfaces.BooleanOutPort.
```

from the library `Modelica.Blocks`.

3.2 Bodies

This category classes were modified to take into account the possibility of impacts in MBSUC. The base class `RigidBody` considered in [1] has been slightly rearranged and now reads as

```
partial model RigidBody
  replaceable KinematicPort OutPort;
  ...
  Real Active(start=1);
equation
  der(Active) = 0;
  der(r) = Active*v;
  der(v) = Active*a;
  der(q) = Active*0.5*QMult(q,
    {0,omega[1],omega[2],omega[3]});
  der(omega) = Active*epsilon;
  ...
end RigidBody;
```

Dots here mean those parts of the `RigidBody` class from the previous version which haven't been reconstructed. In addition, one can see easily that the time of dynamics can be "stopped" here at all. This can be done with auxiliary variable `Active` putting its value equal to zero. In this case the model will be transformed from dynamical to the static one, which is defined by algebraic equations only.
Declaration **`replaceable`** is aimed to provide the possibility of choice between modes of simulation with or without impacts.
To implement impact calculations one uses the following class

```
partial model RigidBodyImpacts
  extends RigidBody(redeclare
    KinematicPortImpacts OutPort);
  SI.Velocity Deltav[3];
  SI.AngularVelocity Deltaomega[3];
  SI.Impulse ImpactForce[3];
  SI.AngularImpulse ImpactTorque[3];
  Boolean Impact;
  SI.Force F1[3];
  SI.Torque M1[3];
  WrenchPortImpacts InPort1;
  BooleanInPort InImpactSignal1;
  BooleanOutPort OutImpactSignal1;
equation
  F = InPort1.F + F1;
  M = InPort1.M + cross(InPort1.P - r,
    InPort1.F) + M1;
  OutImpactSignal1.signal[1] = false;
  Impact = false or
    InImpactSignal1.signal[1];
  OutPort.Deltav = Deltav;
  OutPort.Deltaomega = T*Deltaomega;
  OutPort.Deltav = Deltav;
  OutPort.Deltaomega = T*Deltaomega;
  m*Deltav = InPort1.ImpactForce +
    ImpactForce1;
  I*Deltaomega = transpose(T)*
    (InPort1.ImpactTorque + cross(
    InPort1.P - r,InPort1.ImpactForce)
    + ImpactTorque1);
end RigidBodyImpacts;
```

Since this class is introduced to process impacts then it possesses at least one wrench port supposed for at least one unilateral constraint, which is a potential source of impacts. The class, cited rather its object can be instantiated in the models being developed according to any causality principle. In case of the acausal approach such class is to be completed by the following model

```
model RigidBodyImpactsAcausal
  extends RigidBodyImpacts;
equation
  when Impact then
    reinit(v, v + Deltav);
    reinit(omega, omega + Deltaomega);
  end when;
end RigidBodyImpactsAcausal;
```

providing a self-governing possibility of the object to recalculate velocities at impact. In case of the causal approach such the calculation should be instantiated outside the object of the MBSUC state. Note that in implementations in derived classes for the bodies of MBSUC we can instantiate any number of wrench port objects necessary for constraints of the MBSUC model.

3.3 Constraints

On the same way as for `RigidBody` class the base model `Constraint` has been slightly rearranged and now has the following modified form

```
partial model Constraint
  parameter Integer ConstraintNo = 1;
  replaceable KinematicPort InPortA;
  replaceable WrenchPort OutPortA;
  replaceable KinematicPort InPortB;
  replaceable WrenchPort OutPortB;
equation
  ...
end Constraint;
```

Then one can construct easily the constraint base model taking into account impacts of bodies in the form

```
partial model ConstraintImpacts
  extends Constraint(
    redeclare KinematicPortImpacts
      InPortA,
    redeclare WrenchPortImpacts
      OutPortA,
    redeclare KinematicPortImpacts
      InPortB,
    redeclare WrenchPortImpacts
      OutPortB);
equation
  OutPortA.ImpactForce +
    OutPortB.ImpactForce = zeros(3);
  OutPortA.ImpactTorque +
    OutPortB.ImpactTorque = zeros(3);
end ConstraintImpacts;
```

Now it is time to construct a base model for the unilateral constraint satisfying our assumptions stated earlier and processing impact events correctly

```
model UnilateralConstraintAcausal
  extends ConstraintImpacts;
  parameter Real k;
  parameter Real f;
  parameter SI.Velocity delta;
  UnilateralConstraintState State;
  Boolean Impact;
  Boolean NormalImpact;
  Boolean NormalImpactIndicator;
  Boolean ImpactMask;
  Real[3] normA;
  SI.Impulse ImpactForcen;
  SI.Impulse[3] ImpactForcet;
  SI.Impulse kappa;
  SI.Acceleration[3] arA;
  SI.Acceleration[3] arB;
  SI.Acceleration[3] rela;
  SI.Acceleration relan;
  Real Active(start=1);
  ...
algorithm
  when relan > 0 and not ImpactMask
    then
      ImpactMask := true;
  end when;
  when State == 0 and pre(State) <> 0
    then
      ImpactMask := false;
  end when;
equation
  ...
  NormalImpactIndicator = if mu < 0
    and State == 0 and ImpactMask
    then true else false;
  NormalImpact = edge(
    NormalImpactIndicator);
  Impact = if noEvent(NormalImpact)
    then true else false;
  Active*arA = der(vrA);
  Active*arB = der(vrB);
  rela = arA - arB;
  relan = rela*normA;
  ImpactForcen = OutPortA.ImpactForce*
    normA;
  ImpactForcet = OutPortA.ImpactForce -
    ImpactForcen*normA;
  if noEvent(Impact) then
    if relvtsqrt <= delta then
      zeros(3) = ImpactForcet +
        f*abs(ImpactForcen)*
        relvt/delta - kappa*normA;
    else
      zeros(3) = ImpactForcet +
        f*abs(ImpactForcen)*
        relvt/relvtsqrt - kappa*normA;
    end if;
  else
    zeros(3) = DeltavrAt + vrAt -
      vrBt - DeltavrBt - kappa*normA;
  end if;
  der(Active) = 0;
end UnilateralConstraintAcausal;
```

State of the constraint is tracked here by the variable `State`. If `State = 0` then the constraint is disconnected. For `State = 1` the constraint is in the state “Sliding”. And for `State = 2` corresponding state is “Rolling”. In a current version of the MBSUC model we suppose that at each instant of time it is possible to occur not more than one impact.
Modelica code presented above has the following features:

1. Impact signal is generated if and only if: the con-

straint be in the state "Flight", `State = 0`; outer surfaces of the bodies arrive to the contact, $\mu < 0$; and a special impact mask is open. This latter becomes closed for the only case of the smooth launching from the constraint. The variable μ is defined according to the differential version of the system (2.1) such that for $\mu > 0$ the constraint is disconnected, and the contact begins as $\mu = 0$.

2. Kinematic formulae and expressions for the impact impulses are implemented.

3. The variable `Active` is applied here to scale the independent variable as it has been done for the `RigidBody` model.

4. The following parameters of problem are applied: `k` is the coefficient of restitution at impact, `f` is the friction coefficient, `delta` is the regularizing parameter for dry friction.

Dots represent the blocks of an equations implementing the functions: (a) impact signal transmission through the constraint, now under the further development; (b) computation of an intermediate variables according to formulae (2.2, 2.3, 2.4, 2.7)
A key role in the whole model plays the following class

```
model
UnilateralConstraintAcausalAddOnRegular
  extends UnilateralConstraintAcausal;
  parameter SI.Length
    ClearanceTolerance;
  parameter SI.DampingCoefficient
    ConstraintAttraction=1;
  SI.Force nu;
  SI.Force Forcen;
  SI.Force[3] Forcet;
  SI.Acceleration Drelvn;
  Real StateIndicator;
  SI.Length Clearance(start=1);
equation
  der(relvn) = Active*(Drelvn + (if
    StateIndicator > 0.5 then -
    ConstraintAttraction*relvn else
    0));
  Forcen = OutPortA.F*normA;
  Forcet = OutPortA.F - Forcen*normA;
  if StateIndicator <= 0.5 then
    State = 0;
    Forcen = 0;
    Forcet - nu*normA = zeros(3);
    if mu > 0 and relan < 0 then
      StateIndicator = 0;
    else
      if Clearance < ClearanceTolerance
        then
        if relan < 0 then
          // Case of launch
          StateIndicator = 0;
        else
          // Case of landing
          if relvtsqrt > delta then
            StateIndicator = 1;
          else
            StateIndicator = 2;
          end if;
        end if;
      else
        StateIndicator = 0;
      end if;
    end if;
  else
    Drelvn = 0;
    if relvtsqrt <= delta then
      State = 2;
      StateIndicator = if Forcen > 0
        then 0 else 2;
      Forcet - f*Forcen*relvt/delta -
        nu*normA = zeros(3);
    else
      State = 1;
      StateIndicator = if Forcen > 0
        then 0 else 1;
      Forcet - f*Forcen*relvt/relvtsqrt
        - nu*normA = zeros(3);
    end if;
  end if;
  der(Clearance) = 0;
  when Impact then
    reinit(Clearance, 0.5*abs(relvn*
      relvn/Drelvn));
  end when;
  when StateIndicator > 0.5 then
    reinit(Clearance, 1);
  end when;
  OutPortA.M = zeros(3);
  OutPortA.ImpactTorque = zeros(3);
end
UnilateralConstraintAcausalAddOnRegular;
```

which implements the real switching of the constraint states.
The hybrid automata states are controlled by two variables: `StateIndicator` and `State`. The first one is included into the algebraic loops and has the `Real` type. Hence in some sense the states themselves correspond to fuzzy values and are identified by the inequalities. It is clear that such a situation is connected with the compiler restrictions. The variable `State` doesn't belong to the algebraic loops. It has the `Integer` type and doesn't influence on the switching between

the states.
In frames of the model under consideration in order to estimate the maximal clearance between the bodies during the time from one impact to the next one, the variable `Clearance`, is introduced to detect the instant of the transition to the state of the bodies contact. The complementarity rules are also implemented here.

Remark 3.1 *We have to perform the regularization of the independent variable for the case of landing on the constraint using variables* `Active` *of the bodies and the constraints objects outside these objects but inside the corresponding container of the whole MBSUC model. In this case we have a possibility for the correct control over the regularization process because the change of the independent variable should be total throughout the MBSUC.*

At last, the models implementing the system of constraint equations complete a chain of inheritance for the constraint classes. Namely two classes

```
SurfacesOfConstraintAcausalDifferential,
EllipsoidAndHorizontalPlaneDifferential.
```

have been constructed. First one doesn't depend upon specific type of the outer surfaces. The second model implements a specific case of the ellipsoidal surface and the plane as a surfaces of constraint.

4 Examples

Experimental computations and verification of the models developed were carried out using a well known example from classical dynamics: motion of heavy body on/over the horizontal surface. Visual image of the MBSUC model is presented in Figure 4.1.

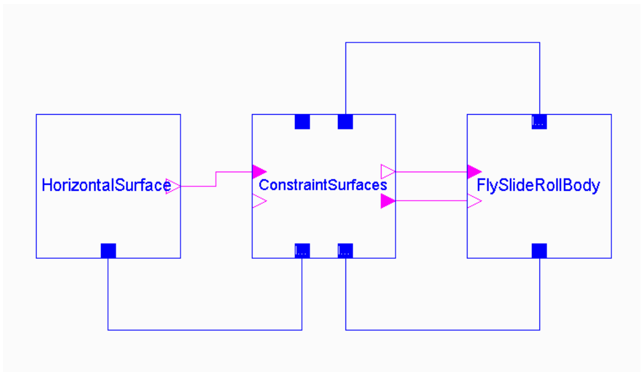

Figure 4.1: Visual Model of MBSUC

The object `HorizontalSurface` on the left hand side of the figure represents model of the base body describing a horizontal plane fixed in *AF*. The object of the heavy rigid body is shown on the right hand side of the Figure 4.1. And the model of total MBSUC for our example has the following Modelica code

```
model MBSAcausalDifferential
  ...
  parameter Period TimeScale=1;
  Period deltat(start=1);
  Time t(start=0);
equation
  ...
  der(deltat) = 0;
  der(t) = deltat/TimeScale;
  when ConstraintSurfaces.Impact then
    reinit(deltat,min(1,2*abs(
      ConstraintSurfaces.relvn/
      ConstraintSurfaces.relan)));
    reinit(FlySlideRollBody.Active,
      min(1,2*abs(ConstraintSurfaces.
      relvn/ConstraintSurfaces.relan)));
    reinit(ConstraintSurfaces.Active,
      min(1,2*abs(ConstraintSurfaces.
      relvn/ConstraintSurfaces.relan)));
  end when;
  when ConstraintSurfaces.
    StateIndicator > 0.5 then
    reinit(deltat, 1);
    reinit(FlySlideRollBody.Active, 1);
    reinit(ConstraintSurfaces.Active,
      1);
  end when;
end MBSAcausalDifferential;
```

To estimate an accuracy of the model developed we performed a comparison of the results with ones for the exact model of the hybrid automata built using causal approach with the three instantiated objects each corresponding to one state of the mechanical system and having a structural complexity of the whole MBSUC, see Figure 4.1.
The rigid body already considered in one of the examples of the paper [1] starts its motion from a position suspended over the surface with the initial data

$$\begin{aligned} \mathbf{r}(0) &= (0,5,0)^T, & \mathbf{v}(0) &= (0.05,0,0)^T, \\ \mathbf{q}(0) &= (1,0,0,0)^T, & \boldsymbol{\omega}(0) &= (0,-10,2)^T. \end{aligned} \qquad (4.8)$$

Motion is simulated on time segment $[t_0,t_1] = [0,150]$ and consists of the several stages of flight alternating by stages of sliding. Note that sliding followed by rolling as energy decreases. During several decades of seconds one can observe easily so called stick–slip phenomena transferring finally to the pure rolling.
The results of simulation are presented in the Figures 4.2, 4.3, 4.4. The final part of the projection of the trajectory for the point P_B of the ellipsoid corresponding to the stick–slip phase is shown in Figure 4.2.

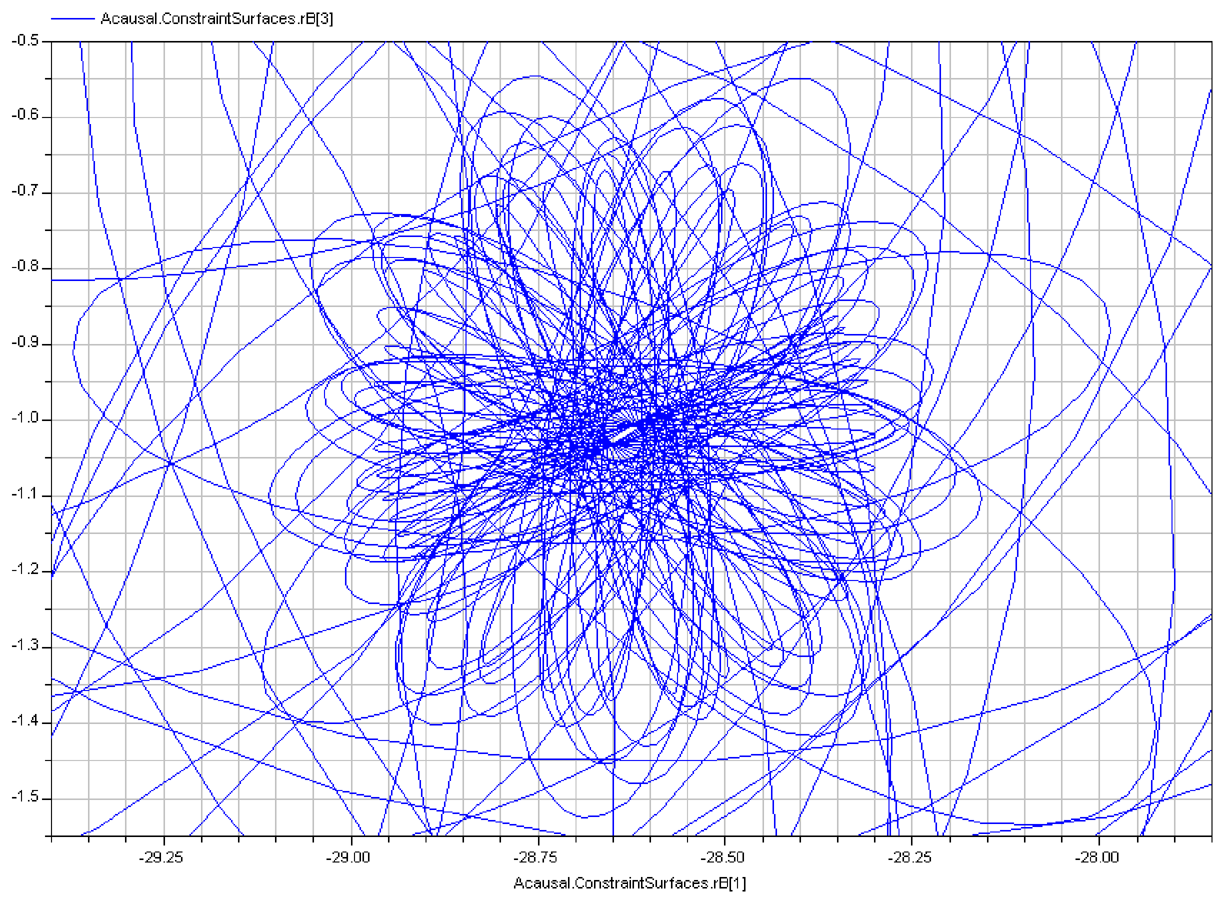

Figure 4.2: Stick–Slip Oscillations, Contact Point Trajectory

An accuracy of the model is of our special interest. The accuracy of computation for instants of impacts and of transitions between the MBSUC states is a causal point for the models of systems with impacts. In our example the instant of the first transfer to rolling at the beginning of stick–slip oscillations has a relative error of the order 10^{-4}. Such an error was accumulated after several thousands of impacts and several transitions between states “Flight” and “Sliding”. More accurate regularization of the independent variable allows to achieve further reduction of the error. Of course it needs considerable computational time in addition. For comparison of physical time variables depending on the regularizing time for the models compared see Figure 4.3. As one can see, physical times almost coincide for the acausal and causal models. In addition, it would be interesting to observe the initial interval of the simulation corresponding to the several stages of a decrementing bouncing of the body, see Figure 4.4. Here we can see the screenshot of the body while it perfoms one of jumps. The image of the fisrt transfer from the flight mode to the mode of sliding is presented here in details. One can see in this inserted fragment the regularizing independent “time” counted along x-axis. y-axis represents the variable μ.

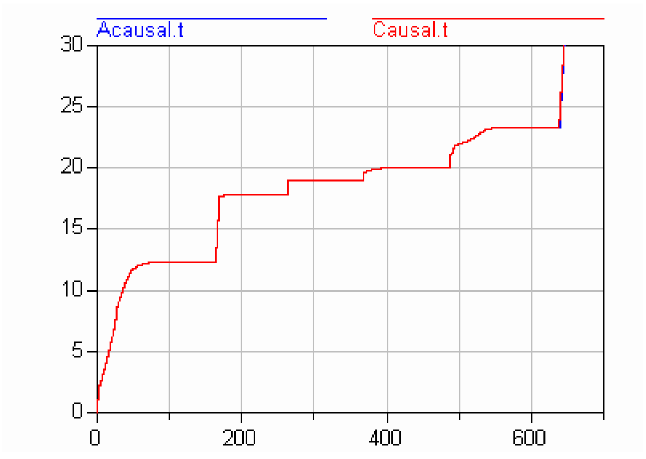

Figure 4.3: Physical Times Depending upon Regularizing Time

In the case of motion with a contact the switching between sliding and rolling is observed. For this case the simulation was performed with the following initial data

$$\mathbf{r}(0) = (0,1,0)^T, \qquad \mathbf{v}(0) = (0.05,0,0)^T,$$
$$\mathbf{q}(0) = (1,0,0,0)^T, \quad \boldsymbol{\omega}(0) = (0,-2,2)^T.$$

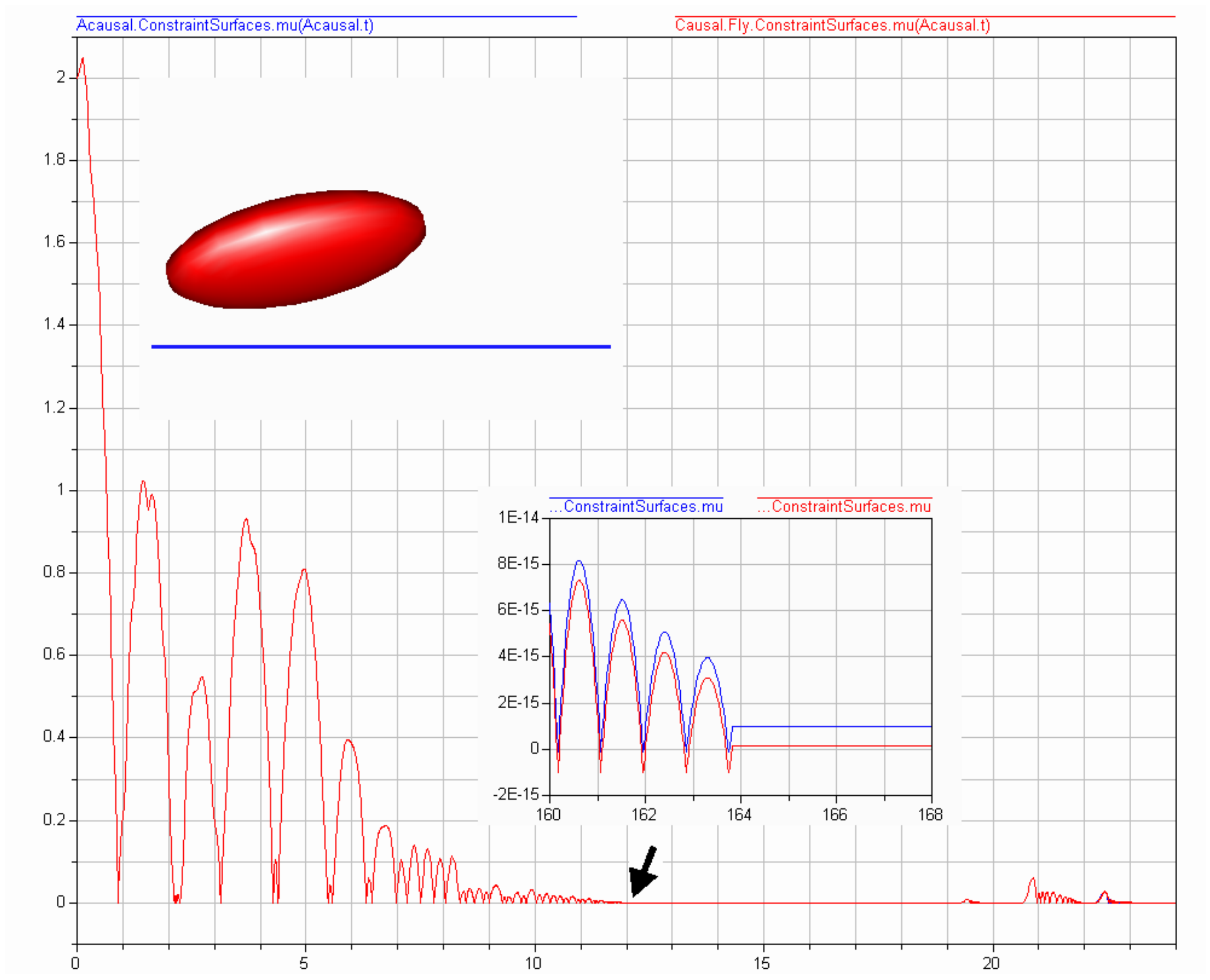

Figure 4.4: Stages of Bouncing

During the time of $t_1 - t_0 = 150$ units and after several hundreds of stick–slip oscillations the relative error accumulated for state switch instants was equal to 10^{-11}. Thus the absence of impacts during the simulation improves the quality of the model more than in million times.

For the sliding/rolling mode the absolute error of determination of the contact point does not exceed $3 \cdot 10^{-5}$. It was observed that the error grows almost linearly. The error in determination of the position of the point P_B in the mormal direction is equal to $2 \cdot 10^{-15}$, while for rolling the error of determination of the tangent component of velocity of this point does not exceed 10^{-7}.

Let us consider now the motion of the homogeneous body bounded by an ellipsoidal surface on the horizontal plane [7]. The coefficient of the Coulomb friction supposed to be equal to $d = 0.01$. Let us try to repeat numerically the following experiment described qualitatively by A. P. Markeev. *The body touches the horizontal surface by its shortest semi–axis at the initial instant. Let us put it in rapid rotation. Then the body tends to the position in which it touches the plane by its longest semi–axis.*

In our example the semiaxes of the body are close one to another: $a_1 = 1.2$, $b_1 = 1$, $c_1 = 1.3$. Axes of outer surface ellipsoid coincide with ones of central principal ellipsoid. Choosing the initial data as in (4.8) with one exception: $\mathbf{r}(0) = (0, 1, 0)^T$ one obtains the result cited above: the ellipsoid masscenter “rises” progressively from the height of minimal semi–diameter to one of maximal semi–diameter, see Figure 4.5. The angular velocity almost holds its direction with respect to the *AF*, see Figure 4.6, blue (lower) curve. At the initial instant this vector is directed along the minimal semi–axis, red (middle) curve is its projection on the corresponding axis of the body; while on the final stage the angular velocity is directed along the maximal ellipsoid semi–axis, green (upper) curve.

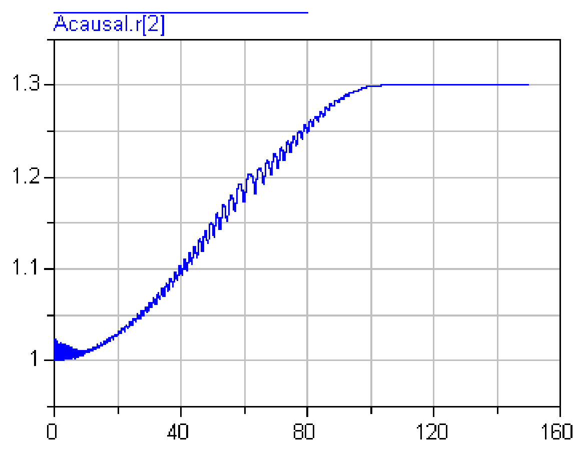

Figure 4.5: Center of Mass Altitude

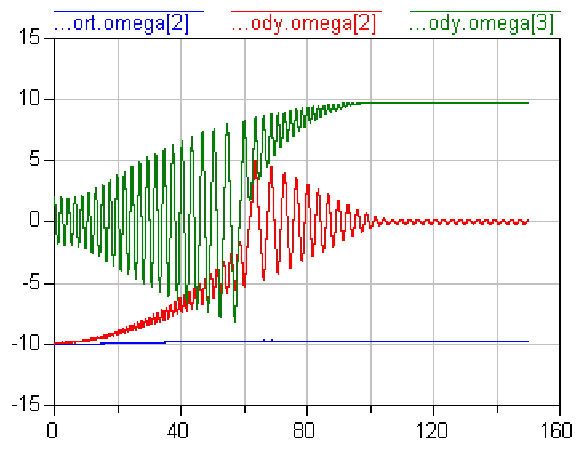

Figure 4.6: Projections of Angular Velocity

5 Conclusions

Summurizing the results obtained while developing the class library for the dynamics of the MBSUC let us enumerate several relevant problems and their solutions.

Problem 1: How one can implement the geometry of the unilateral constraint? **Solution:** Use the system of algebraic equations like (2.1).

Problem 2: How one can ensure the reliability of the implementation of the constraint? **Solution:** Use the differential form of the equations (2.1).

Problem 3: How one can implement impacts in MBSUC in the acausal manner? **Solution:** Use the independed algebraic subsystem of equations distributed throughout the MBSUC model.

Problem 4: How one can implement the dichotomy flight/contact? **Solution:** Use the complementarity rule for the normal force of reaction and the derivative of the normal relative velocity at the contact point.

Problem 5: How one can implement the dichotomy slipping/rolling? **Solution:** Use the regularized tangent force for the Coulomb friction.

Problem 6: How one can implement the exact “landing” on the constraint? **Solution:** Use the regularizing independent variable for the total model.

Problem 7: How one can implement switching between states of the constraint in the acausal manner? **Solution:** Use the **if** clause in combination with the state variable of `Real` type. This variable is included to corresponding algebraic loop. As a result the structural complexity of the total model doesn’t increase.

6 Acknowledgement

The paper was prepared with partial support of Russian Foundation for Basic Research, grants 02–01–00196, SS–2000.2003.1.

References

[1] Kossenko, I. I., and Stavrovskaia, M. S., How One Can Simulate Dynamics of Rolling Bodies Via Dymola: Approach to Model Multibody System Dynamics Using Modelica // Proceedings of the 3rd International Modelica Conference, Linköpings universitet, Linköping, Sweden, November 3–4, 2003, pp. 299–309.

[2] Pfeiffer, F., Unilateral Multibody Dynamics // Meccanica, 1999, Vol. 34, No. 6, pp. 437–451.

[3] Novozhilov, I. V., Fractional Analysis : Methods of Motion Decomposition. — Boston: Birkhauser, 1997.

[4] Routh, E. J., A Treatise on the Dynamics of a System of Rigid Bodies. — London: Vol. 1, 1897.

[5] Ivanov, A. P., Dynamics of Systems with Mechanical Impacts. — Moscow: 1997. ISBN 5-7781-0031-0.

[6] Dymola. Dynamic Modeling Laboratory. User’s Manual. Version 5.1b — Lund: Dynasim AB, Research Park Ideon, 2003.

[7] Markeev, A. P., On the Motion of an Ellipsoid on a Rough Surface with Slippage. // Journal of Applied Mathematics and Mechanics, Vol. 47, Iss. 2, 1983, pp. 260–268.