Event Handling Solver Compared to Modelica Dassl

Boucing Ball Benchmark

T.H. Gallois (IFPEN), J. Brac (IFPEN), T. Soriano (SUPMECA)
Outline

■ Introduction and problem definition
■ Benchmark
■ The constraint vector
■ Computation of the event time $t^*$
■ Computation of the solution at $t^*$
■ Results for hard spheres
■ Results for soft spheres
■ Conclusion
Outline

- Introduction and problem definition
- Benchmark
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
- Conclusion
Introduction and problem definition

- What is a **hybrid system**?

- What are the **type of events**?
  - predictable events
  - unpredictable events

- What are the **types of model modifications after an event**?
  - change of initial conditions
  - change of equations
  - change of the state vector
Outline

- Introduction and problem definition
- Benchmark
  - Dynamic equations
  - About the numerical scheme
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
- Conclusion
Outline

- Introduction and problem definition
- Benchmark
  - Dynamic equations
  - About the numerical scheme
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
  - Constraint vector
  - System of equations
  - Results
- Conclusion
Benchmark: The Bouncing Ball

Dynamic Equations of the bouncing ball with air resistance

Air resistance: \( F_{\text{drag}} = -\frac{1}{2} C_x \rho_{\text{air}} S ||v||v \)

\[
\begin{align*}
    m \frac{d}{dt} v_x &= -\frac{1}{2} C_x \rho_{\text{air}} S ||v||v_x \\
    m \frac{d}{dt} v_y &= -m \ g - \frac{1}{2} C_x \rho_{\text{air}} S ||v||v_y \\
    \frac{dx}{dt} &= v_x \\
    \frac{dy}{dt} &= v_y
\end{align*}
\]

Initial conditions

\[
\begin{align*}
    v_x(t = 0) &= v_{x0} & v_y(t = 0) &= v_{y0} \\
    x(t = 0) &= x_0 & y(t = 0) &= y_0
\end{align*}
\]
Outline

■ Introduction and problem definition

■ Benchmark
  ■ Dynamic equations
  ■ About the numerical scheme

■ The constraint vector

■ Computation of the event time $t^*$

■ Computation of the solution at $t^*$

■ Results for hard spheres

■ Results for soft spheres
  ■ Constraint vector
  ■ System of equations
  ■ Results

■ Conclusion
About the numerical scheme

**Numerical scheme:** Backward Differentiation Formula of high order
→ good stability

**After each event:** Implicit Runge-Kutta scheme
→ A-stable, allows to adapt the time step at each restart of the solver
This small difference before the first bounce will be bigger and bigger after many bounces.
The constraint vector

- Hybrid dynamic systems have several constraints. All these constraints are the component of a vector $k$.
- The sign change of a constraint triggers the occurring of an event.
- In our example $k$ is a scalar and $k = y - y_{ground} \geq 0$

\[
\begin{align*}
    m \frac{d}{dt} v_x &= -\frac{1}{2} C_x \rho_{air} S ||v|| v_x \\
    m \frac{d}{dt} v_y &= -mg - \frac{1}{2} C_x \rho S ||v|| v_y \\
    \frac{dx}{dt} &= v_x & \frac{dy}{dt} &= v_y \\
    k &= y - y_{ground} \geq 0
\end{align*}
\]

Initial conditions

\[
\begin{align*}
    v_x(t = 0) &= v_{x0} & v_y(t = 0) &= v_{y0} \\
    x(t = 0) &= x_0 & y(t = 0) &= y_0
\end{align*}
\]
Outline

- Introduction and problem definition
- Benchmark
- The constraint vector
- Computation of the event time \( t^* \)
- Computation of the solution at \( t^* \)
- Results for hard spheres
- Results for soft spheres
- Conclusion
Computation of the event time $t^*$

Difficulties

- The numerical integrator is **discretized in time** → we have the solution of the problem only at the points of the integration.
- The challenge is to compute the **instant** when the ball hits the ground.
Computation of the event time $t^*$

Interpolation

- The idea is to interpolate a second degree time polynomial at $y_{n+1}, y_n, y_{n-1}$

- Compute analytically the roots and choose the appropriate one.
- We get the time $t^*$ when the sign of the constraint changes.
Outline

- Introduction and problem definition
- Benchmark
- The constraint vector
- Computation of the event time $t^*$
- **Computation of the solution at $t^*$**
- Results for hard spheres
- Results for soft spheres
- Conclusion
Computation of the solution at time $t^*$

Interpolation

- Now we have $t^*$ we need to compute the solution at this time.

- If the ODE system is rewritten as $\dot{X} = f(X)$ and by considering $X$ the state vector

\[ X = \begin{pmatrix} v_x \\ v_y \\ x \\ y \end{pmatrix}. \]

The solver gives the solution $X_n$ at a given time $t_n$ and $X_n = \begin{pmatrix} v_{xn} \\ v_{yn} \\ x_n \\ y_n \end{pmatrix}$

The idea is to interpolate a third degree time polynomial for each component of the state vector by considering $X_n, X_{n+1}, f(X_n), f(X_{n+1})$.
Computation of the solution at time $t^*$

Interpolation

- Evaluate this polynomial at $t^*$ to get $X^* = X(t^*)$.
- $X^*$ is used afterwards as an initial condition.
Solving the system after an event

- In hard spheres context after each event (contact with the ground) the equations do not change.
- Only the initial conditions of the vertical speed component are modified and have the opposite value at \( t^* \) multiplied by a damping factor \( 1 - \epsilon \).
- Then we get the following system after the ball hits the ground:

\[
\begin{align*}
    m \frac{d}{dt} v_x &= -\frac{1}{2} C_x \rho_{air} S ||v|| v_x \\
    m \frac{d}{dt} v_y &= -mg - \frac{1}{2} C_x \rho_{air} S ||v|| v_y \\
    \frac{dx}{dt} &= v_x \\
    \frac{dy}{dt} &= v_y \\
    c &= y > 0
\end{align*}
\]

New initial conditions

\[
\begin{align*}
    v_x(t = t^*) &= v_x^* \\
    v_y(t = t^*) &= -v_y^*(1 - \epsilon) \\
    x(t = t^*) &= x^* \\
    y(t = t^*) &= y^*
\end{align*}
\]
Results on a flat and sinusoidal ground

On a flat ground:

\[ y(t) \]

\[ x(t) \]
Comparison with DASSL

The bouncing ball model is implemented in Modelica code:

```model BouncingBall
    Real vx(start = 1);
    Real vy(start = 5);
    Real x(start = 0);
    Real y(start = 2);
    parameter Real m = 1.1;
    parameter Real Cx = 0.5;
    parameter Real rho = 1.293;
    parameter Real S = 3.14 * 0.1 * 0.1;
    constant Real g = 9.81;

equation
    m * der(vx) = -0.5 * Cx * rho * S *
        sqrt(vx ^ 2 + vy ^ 2) * vx;
    m * der(vy) = -m * g - 0.5 * Cx * rho * S *
        sqrt(vx ^ 2 + vy ^ 2) * vy;
    der(x) = vx;
    der(y) = vy;
    when y <= 0 then
        reinit(vy, -0.9 * pre(vy));
    end when;
end BouncingBall;
```
Comparison of the CPU time for the simulation of the bouncing ball during 20 seconds with DASSL and our own solver.

- Relative tolerance: $10^{-6}$
- Coefficient of air friction: $C_x \rho S = 0.0203 \text{ kg/m}$.

<table>
<thead>
<tr>
<th>CPU time DASSL</th>
<th>CPU time our solver</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.068s</td>
<td>0.032s</td>
</tr>
</tbody>
</table>
Precision comparison

![Graph showing precision comparison]

The graph illustrates the comparison between two simulation methods, tdassl and tsolver, with their respective times: tdassl 9.12049 s and tsolver 9.12052 s.
Absolute difference for times of events DASSL/solver

![Graph showing absolute difference for times of events DASSL/solver against number of bounces. The x-axis represents the number of bounces ranging from 0 to 12, while the y-axis shows absolute difference in seconds ranging from 0 to 0.00003. The graph shows a consistent increase in absolute difference as the number of bounces increases.]
Comparison of DASSL and our solver with the analytical solution:

Table: Comparison of the time of the first event.

<table>
<thead>
<tr>
<th>number of bounce</th>
<th>relative tolerance</th>
<th>analytical solution</th>
<th>DASSL</th>
<th>error DASSL</th>
<th>our solver</th>
<th>error our solver</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>$10^{-6}$</td>
<td>0.640714</td>
<td>0.640715</td>
<td>$1.56 \times 10^{-6}$</td>
<td>0.640714</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td>0.640714</td>
<td>0.641162</td>
<td>$7 \times 10^{-4}$</td>
<td>0.640730</td>
<td>$2.50 \times 10^{-5}$</td>
</tr>
<tr>
<td>2nd</td>
<td>$10^{-6}$</td>
<td>1.769519</td>
<td>1.769523</td>
<td>$2.26 \times 10^{-6}$</td>
<td>1.769515</td>
<td>$2.26 \times 10^{-6}$</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td>1.769519</td>
<td>1.769915</td>
<td>$2.23 \times 10^{-4}$</td>
<td>1.769562</td>
<td>$2.43 \times 10^{-5}$</td>
</tr>
</tbody>
</table>
Results on a flat and sinusoidal ground

On a sinusoidal ground:

With a small amplitude

With a big amplitude
Outline

- Introduction and problem definition
- Benchmark
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres

Results for soft spheres
- Constraint vector
- System of equations
- Results
Outline

- Introduction and problem definition
- Benchmark
  - Dynamic equations
  - About the numerical scheme
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
  - Constraint vector
  - System of equations
  - Results
- Conclusion
Results in soft spheres context

Constraint vector

Now the ball is supposed to be **deformed** during the contact as:

As the ball is now soft we have to consider now its **radius**, and so the constraint vector becomes:

\[ \mathbf{c} = [y(t) - R] \]
Outline

- Introduction and problem definition
- Benchmark
  - Dynamic equations
  - About the numerical scheme
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
  - Constraint vector
  - System of equations
  - Results
- Conclusion
System of equations during the contact

During the contact the equations of the model change.

We consider now the elastic force: $F_{\text{elas}} = 2\pi E\sqrt{2R} |R - y|^{3/2} u_y$

and the dissipative force: $F_{\text{diss}} = -\mu v$.

So after determining $t^*$ et $X^*$ we have the following system:

\[
\begin{align*}
    m \frac{d}{dt} v_x &= -\mu v_x \\
    m \frac{d}{dt} v_y &= -mg + 2\pi E\sqrt{2R} |R - y|^{3/2} - \mu v_y \\
    \frac{dx}{dt} &= v_x \\
    \frac{dy}{dt} &= v_y \\
    c &= y - R
\end{align*}
\]

Initial conditions

\[
\begin{align*}
    v_x(t = t^*) &= v_{x*} \\
    v_y(t = t^*) &= v_{y*} \\
    x(t = t^*) &= x^* \\
    y(t = t^*) &= y^*
\end{align*}
\]
Outline

- Introduction and problem definition
- Benchmark
  - Dynamic equations
  - About the numerical scheme
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
  - Constraint vector
  - System of equations
- Results
- Conclusion
Results for soft spheres

After the event "the ball hits the ground" we switch to another model described by the previous system. When the ball does not touch the ground anymore there is another event and we go back to the original model.
model SoftBouncingBall

Real \( v_x \) (start = 1);
Real \( v_y \) (start = 5);
Real \( x \) (start = 0);
Real \( y \) (start = 2);

parameter Real \( C_x = 0.5 \);
parameter Real \( \rho = 1.293 \);
parameter Real \( \pi = 3.141592653 \);
parameter Real \( S = \pi \times 0.1 \times 0.1 \);
parameter Real \( R = 0.1 \);
parameter Real \( E = 0.1 \times 10^9 \);
parameter Real \( \text{density} = 500 \);
parameter Real \( m = \frac{4}{3} \pi R^3 \text{density} \);
constant Real \( g = 9.81 \);

equation

if \( y > R \) then

\[ m \times \frac{\text{d}v_x}{\text{d}t} = -0.5 \times C_x \times \rho \times S \times \sqrt{v_x^2 + v_y^2} \times v_x; \]
\[ m \times \frac{\text{d}v_y}{\text{d}t} = -m \times g - 0.5 \times C_x \times \rho \times S \times \sqrt{v_x^2 + v_y^2} \times v_y; \]
\[ \frac{\text{d}x}{\text{d}t} = v_x; \]
\[ \frac{\text{d}y}{\text{d}t} = v_y; \]

else

\[ m \times \frac{\text{d}v_x}{\text{d}t} = 0; \]
\[ m \times \frac{\text{d}v_y}{\text{d}t} = -m \times g + 2 \times 3.14 \times E \times \sqrt{2R} \times \text{abs}(R-y)^{(3/2)}; \]
\[ \frac{\text{d}x}{\text{d}t} = v_x; \]
\[ \frac{\text{d}y}{\text{d}t} = v_y; \]
end if;
end SoftBouncingBall;
Speed comparison

Let us compare the CPU time for the simulation of the soft bouncing ball during 10 seconds with DASSL and our own solver. relative tolerance : $10^{-6}$

<table>
<thead>
<tr>
<th>CPU time DASSL</th>
<th>CPU time our solver</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.061s</td>
<td>0.045s</td>
</tr>
</tbody>
</table>
Precision comparison

Let us look both results for the tenth bounce around 7.51 s
Now we plot the absolute difference of the event times between DASSL and our own solver for the first 10 bounces.
Absolute difference for times of events DASSL/solver

<table>
<thead>
<tr>
<th>Number of Bounces</th>
<th>Absolute Difference (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.0001</td>
</tr>
<tr>
<td>4</td>
<td>0.0002</td>
</tr>
<tr>
<td>6</td>
<td>0.0003</td>
</tr>
<tr>
<td>8</td>
<td>0.0004</td>
</tr>
<tr>
<td>10</td>
<td>0.0005</td>
</tr>
<tr>
<td>12</td>
<td>0.0006</td>
</tr>
</tbody>
</table>

The graph shows the absolute difference in times of events for different numbers of bounces using the DASSL/solver.
Outline

- Introduction and problem definition
- Benchmark
- The constraint vector
- Computation of the event time $t^*$
- Computation of the solution at $t^*$
- Results for hard spheres
- Results for soft spheres
- Conclusion
Conclusion

■ Objectives realized :
  ■ Build an example of model with different types of events
    → change of only initial conditions : hard spheres
    → change of model : soft spheres
  ■ Propose a new method of event handling
    → computation of event time $t^*$
    → computation of the solution $X^*$ at $t^*$

■ Perspectives
  ■ Consider many balls to get a multi-events model which needs to add
    a new model of collision between balls
  ■ Consider the rotation of the ball on itself and the Magnus effect
  ■ Add this benchmark to the standard library of OpenModelica in
    order to share it with the community