Point Abstractions in Continuous Models

Rüdiger Hohmann*

Abstract

Point abstractions are singularities in a continuum. They can be described by delta functions. The given method puts prior to the numeric integration an analytic integration step which transforms the unlimited delta functions into finite step functions. In modern simulation systems discrete time- and state events can generate the step function in view of the numeric integration. Examples investigated from the mechanics are a single force acting down on a heavy sagging rope and the force impulse occuring upon the collision of two masses.

Key words: delta function, preceded analytical integration, step function, discrete events, single force, collisions.

AMS subject classifications: 34A36, 34A37, 65L10, 70E18, 70E55, 70G10, 70G60.

1 Introduction

For potential users in natural sciences and technology, a methodical approach for the inclusion of point abstractions in continuous models has to be provided. Point abstractions, also called sharp concentrated quantities, are deemed physical abstractions, like the point mass, a single force in a finite force density, or a force impulse in vanishing time. They are singularities in a continuum. The delta function represents a suitable form to describe such point abstractions. They are substituted for this purpose by means of a δ_ε-function with a finite distribution (density) and, after that, the corresponding differential equation is integrated analytically. The integration starts at that point where the initial conditions are given. This way they remain relevant, and the consistency with the other differential equations is unchanged. Only the highest derivative and terms with δ_ε-functions have to be dealt with analytically. The remaining terms are integrated numerically in the course of the simulation. Now, it is allowed to carry out the limiting process $\varepsilon \to 0$ in such terms where the δ_ε-function is below an integral. This way the point abstractions are created. Step functions result therefrom which can be represented easily in continuous simulation systems.

2 Integration of model equations

The method for the modelling of point abstractions puts prior to the numeric integration an analytical integration step of those differential equations that contain delta functions, which this way turn into finite step functions. Then they could be generated and integrated numerically by means of discrete events. This combination of analytical with numerical integration is characteristic for this method.

Independent of the mathematical form and independent of ε, the δ_ε-functions have the characteristic feature that their integral in the interval from -∞ to +∞, equals one. With decreasing values of ε they get narrower and higher. The term

$$\delta_\varepsilon(x-a) = \begin{cases} 1/\varepsilon & \text{for} \quad a-\varepsilon/2 \le x \le a+\varepsilon/2 \qquad \text{with } \varepsilon > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

*Otto-von-Guericke-Universität Magdeburg, Institut für Simulation und Graphik, Universitätsplatz 2, 39106 Magdeburg, Germany, hohmann@isg.cs.uni-magdeburg.de

16th IMACS World Congress (© 2000 IMACS)

is the *Dirac* δ_ε-function at $x = a$ in the narrow sense. It is not permitted to set ε equal zero, but for continuous functions $y(x)$ exist the limit of the integral [7]

$$\lim_{\varepsilon \to 0} \int_{-\infty}^{+\infty} \delta_\varepsilon(x' - x) y(x')\, dx' = y(x). \tag{2}$$

As a way of shortening the function δ_ε(x) and the limiting process $\varepsilon \to 0$, permitted subsequent to integrations only, the **symbol $\delta(x)$ is defined**. In nonintegrated equations it corresponds with terms in the form of (1). Below integrals, however, the symbol has to be substituted by

$$\int \delta(x' - x) y(x')\, dx' \equiv \lim_{\varepsilon \to 0} \int \delta_\varepsilon(x' - x) y(x')\, dx'. \tag{3}$$

Model equation with δ-function at $t = t_1$ of a dynamic system (IVP) [5]:

$$\begin{aligned} y^{(n)} &= f(t,\, y,\, \dot{y},\, \ldots,\, y^{(n-1)}) + g(t)\delta(t - t_1), \\ y(0) &= y_0,\, \dot{y}(0) = \dot{y}_0,\, \ldots,\, y^{(n-1)}(0) = y_0^{(n-1)}. \end{aligned} \tag{4}$$

By means of analytical integration, within the range from 0 to t, we get

$$y^{(n-1)} = \int_0^t f(t',\, y,\, \dot{y},\, \ldots,\, y^{(n-1)})\, dt' + g(t_1)s(t - t_1) + y_0^{(n-1)}, \tag{5}$$

with the unit step function

$$s(t - t_1) = \begin{cases} 0 & \text{for } t < t_1 \\ 1 & \text{for } t > t_1, \end{cases} \tag{6}$$

as it is called in the time domain. The function $g(t)$ is supposed to be continuous:

$$\int_0^t \delta(t' - t_1) g(t')\, dt' = g(t_1)s(t - t_1) = \begin{cases} 0 & \text{for } t < t_1 \\ g(t_1) & \text{for } t > t_1. \end{cases} \tag{7}$$

The unit step function $s(t - t_1)$ results in values jumping between 0 and 1 by integration of a delta function with coefficient function $g(t) = 1$.

Now, $y^{(n-1)}$ can be calculated numerically. The simulation system ACSL [1] allows a direct notation of (5) by means of the integration block INTEG:

```
y(n-1) = INTEG(f, y(n-1)0)+g(t1)*s(t-t1).
```

Using an explicit representation of the integral

$$u = \int_0^t f(t',\, y,\, \dot{y},\, \ldots,\, y^{(n-1)})\, dt' + y_0^{(n-1)}, \tag{8}$$

we can also specify a system consisting of two differential equations of first and $(n-1)$th order for the time functions $u(t)$ and $y(t)$:

$$\begin{aligned} \dot{u} &= f(t,\, y,\, \dot{y},\, \ldots,\, y^{(n-1)}), & u(0) &= y_0^{(n-1)}, \\ y^{(n-1)} &= u + g(t_1)s(t - t_1), & y(0) &= y_0,\, \dot{y}(0) = \dot{y}_0,\, \ldots,\, y^{(n-2)}(0) = y_0^{(n-2)}. \end{aligned} \tag{9}$$

Before the point abstraction has an effect on $t = t_1$, the functions $u(t)$ and $y^{(n-1)}(t)$ correspond and they differ from each other after that by a constant value $g(t_1)$.

3 Step functions

In modern continuous simulation systems capable of handling time- and state dependent discrete events, step functions obtained by means of integration of a delta function can be generated elegantly. We distinguish between time and state events. The step point at time events is given a priori, meanwhile it has to be iterated at state events. The zero crossing of an arithmetic term initiates a state event in ACSL. A DISCRETE section will be called by the SCHEDULE statement in order to change the step variable. The integration intervals get their limits out of an event list. This way the numerical integration methods do not pass over these limits, but they stop here and begin new runs in the following interval. Errors by discontinuities are ruled out, as a consequence.

Program sequences for time events in ACSL (selected statements):

(a)

```
...
INITIAL
s = 0.0
SCHEDULE Event .AT. t1
END ! of initial
...
DERIVATIVE
Model equations
END ! of derivative

DISCRETE Event
CALL LOGD(.TRUE.)
s = 1.0
CALL ZZDERV(1)
CALL LOGD(.TRUE.)
END ! of event
...
```

(b)

```
...
INITIAL
g1s = 0.0
SCHEDULE Event .AT. t1
END ! of initial
...
DERIVATIVE
Model equations
END ! of derivative

DISCRETE Event
CALL LOGD(.TRUE)
g1s = g
CALL ZZDERV(1)
CALL LOGD(.TRUE.)
END ! of event
...
```

Case (a) generates the unit step, while the step height in case (b) amounts to $g(t_1)$. Here the δ-function has a coefficient function $g(t)$ according to (4). The call ZZDERV(1) after a change of the step function value at $t = t_1$ causes an evaluation of all model equations. By calls of LOGD before and after the change of $s = s(t - t_1)$ and $g1s = g(t_1)s(t-t_1)$, respectively, the resulting two value sets of selected model variables are logged on an intermediate data file. This permits the exact reproduction of the discontinuity in a graphical representation.

If the time of the event is not given at the beginning, but it depends on the course of the dynamic process, the SCHEDULE operator can also iterate this time as a state event. To this end an arithmetic term is defined, whose zero crossing generates the event. The forms of the SCHEDULE statement applicable to state events are .XZ. (zero crossing in both directions), .XP. (positive crossing), .XN. (negative crossing). The statement is now placed in the DERIVATIVE section together with the model differential equations. The symbol X stands for crossing.

As an example, the collision of two masses with the inner position co-ordinates $x_1(t)$ and $x_2(t)$ is to cause a state event. At the beginning, the co-ordinates shall be $x_1 < x_2$ and the velocities $v_1 > v_2$. Then a possible SCHEDULE statement could be

```
SCHEDULE x1-x2 .XP. Collision.
```

After the iteration process has terminated in a minimal deformation, determined by the error limit, the DISCRETE section "Collision" is called in which the reactions are implemented. The model time is stopped during the calculation (collision process without finite period of time).

4 Examples

Single forces acting on a heavy rope which causes a finite continuous force density by its own weight, and the impact forces between two masses can be described with delta functions. These quantities appear explicitly in the differential equations, however they are eliminated in the classical description and substituted by fitting conditions between separate intervals. Their influence is here no longer obvious from the equations. For this reason, the delta function is a suitable, problem-near option for description and modelling in simulation.

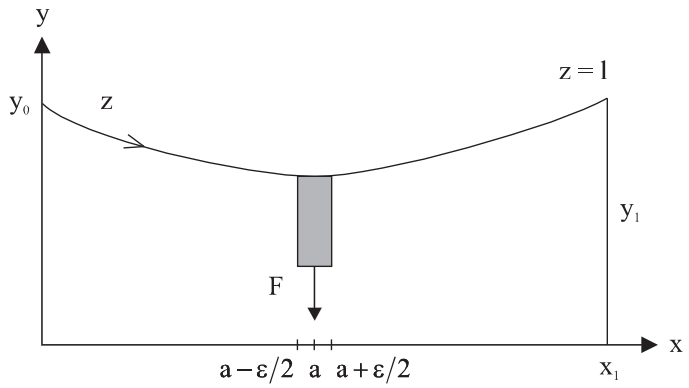

Figure 1: A single force acts down on a sagging rope by means of a band.

4.1 Single force on heavy rope

A heavy sagging rope of length l is fastened between two poles of the height y_0 and y_1 at places $x = 0$ and $x = x_1$. An additional single force at $x = a$ $(0 < a < x_1)$ acts down on the rope. Further quantities needed are the mass/length of rope ρ, the acceleration of gravity g, the constant horizontal rope force component H, and the outer force density $q(x)$ [2].

The boundary conditions in the equations of the rope curve $y(x)$ and the arc length $z(x)$ are satisfied by suitable values of $y'(0) = y'_0$ and H:

$$y'' = \frac{\rho g}{H}\sqrt{1 + y'^2} + \frac{q(x)}{H}, \qquad y(0) = y_0, \quad y(x_1) = y_1, \tag{10}$$

$$z' = \sqrt{1 + y'^2}, \qquad z(0) = 0, \quad z(x_1) = l. \tag{11}$$

The single force F here generates the force density

$$q(x) = F\,\delta(x - a), \tag{12}$$

that is, F is distributed before the limiting process on the neighbourhood of $x = a$ as finite force density. The δ_ε-function (1) would describe the case that a band of the width ε is located symmetrically to $x = a$ on the rope, by which hangs a weight F (Figure 1). During the passing to the limit $\varepsilon \to 0$ the band passes over in a "tiny thin" wire. After the integration of y'' and the limiting process results the first order system of differential equations:

$$u' = \frac{\rho g}{H}\,z', \qquad u(0) = y'_0, \tag{13}$$

$$y' = u + \frac{F}{H}\,s(x - a), \qquad y(0) = y_0, \quad y(x_1) = y_1, \tag{14}$$

$$z' = \sqrt{1 + y'^2}, \qquad z(0) = 0, \quad z(x_1) = l. \tag{15}$$

The transition of the independent default variable t to the position co-ordinate x takes place in ACSL by means of a VARIABLE statement. In the terminology of this simulation language, the step at $x = a$ remains a time event.

By means of an iteration method, the unknown parameters $p_1 := y'_0$ and $p_2 := H$ need to be determined so that the two functions

$$\begin{aligned} f(p_1, p_2) &:= y(x_1) - y_1, \\ g(p_1, p_2) &:= z(x_1) - l \end{aligned} \tag{16}$$

will be zero simultaneously. The iteration was here executed by a two-dimensional *Newton* method [4]. To this end, the two functions f and g are linearized in the neighbourhood of the approximation (p_{1i}, p_{2i}), and a system of two linear equations has to be solved in order to obtain the next approximation (p_{1i+1}, p_{2i+1}). The rope curve with break point, the accompanying arc length and their discontinuous derivatives are represented in [5, 6].

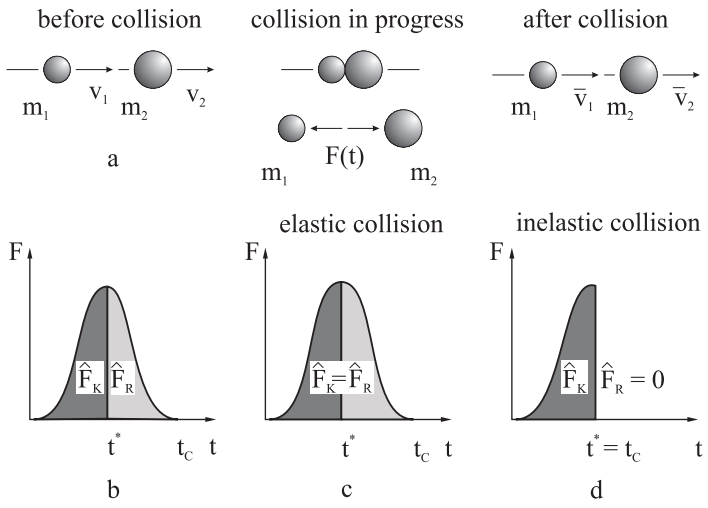

Figure 2: The force impulses during different elastic collision processes.

4.2 Collision of two masses

The collision process takes place in two phases (Figure 2) [3]. The initial compression phase ends with maximum deformations at $t = t^*$. Both bodies have now the same velocity v^*. A force impulse $\hat{F}_C$ is exerted over this first period of time. In the following restitution phase in which a force impulse $\hat{F}_R$ is exerted, the deformations go back partially or completely. The force impulse during the restitution is zero (Figure 2d, inelastic), equal to (Figure 2c, elastic) or lower than (Figure 2b, partial elastic) the one during the compression:

$$\hat{F}_R = e\,\hat{F}_C, \tag{17}$$

with $0 \le e \le 1$ (e restitution coefficient).

The force impulse $\hat{F}$ exerted altogether over the period of the collision time t_c is the total of both components, calculated by an integral of the collision force $F(t)$:

$$\hat{F} = \hat{F}_C + \hat{F}_R = \int_0^{t_c} F(t)\,dt. \tag{18}$$

After application of the momentum conservation law at $t = t^*$ results

$$\hat{F} = (1+e)\frac{m_1 m_2}{m_1 + m_2}\,(v_1 - v_2), \tag{19}$$

with the velocities v_1, v_2 immediately before the collision (Figure 2a) and $\overline{v}_1$, $\overline{v}_2$ after that (*Newton*'s collision law). Definition of a δ_ε-function with $\varepsilon = t_c$:

$$F(t) \equiv \hat{F}\,\delta_\varepsilon(t). \tag{20}$$

An analytical term of $\delta_\varepsilon(t)$ using the variable ε we get by means of a linear transformation [6]

$$\delta_\varepsilon(t) = \frac{t_c}{\varepsilon}\frac{F\left(\frac{t_c}{\varepsilon}t\right)}{\hat{F}}, \tag{21}$$

which turns into the definition equation (20) through $\varepsilon = t_c$. The passing to the limit $\varepsilon \to 0$ describes an impact without a finite period of collision time. This way the collision can be represented as a state event.

Equations of motions in case of multiple impacts at time points t_i and with force impulses $\hat{F}_i$:

$$\begin{aligned} m_1\,\ddot{x}_1 &= F_1(t,\,x_1,\,\dot{x}_1) - \textstyle\sum_i \hat{F}_i\,\delta(t-t_i), \qquad x_1(0) = x_{10}, \quad \dot{x}_1(0) = \dot{x}_{10}, \\ m_2\,\ddot{x}_2 &= F_2(t,\,x_2,\,\dot{x}_2) + \textstyle\sum_i \hat{F}_i\,\delta(t-t_i), \qquad x_2(0) = x_{20}, \quad \dot{x}_2(0) = \dot{x}_{20}. \end{aligned} \tag{22}$$

The time points t_i are determined by the algebraic equation

$$x_1(t_i) - x_2(t_i) = 0, \tag{23}$$

with the inner position co-ordinates x_1, x_2 of the bodies. After integration and limiting process, the equations of motions change into a first order system of four differential equations:

$$\dot{u}_1 = \frac{F_1(t,\, x_1,\, \dot{x}_1)}{m_1}, \qquad u_1(0) = \dot{x}_{10}, \tag{24}$$

$$\dot{x}_1 = u_1 - \frac{1}{m_1} \sum_i \hat{F}_i\, s(t - t_i), \qquad x_1(0) = x_{10}, \tag{25}$$

$$\dot{u}_2 = \frac{F_2(t,\, x_2,\, \dot{x}_2)}{m_2}, \qquad u_2(0) = \dot{x}_{20}, \tag{26}$$

$$\dot{x}_2 = u_2 + \frac{1}{m_2} \sum_i \hat{F}_i\, s(t - t_i), \qquad x_2(0) = x_{20}. \tag{27}$$

Here the force impulses at $t = t_i$ have to be summed up. They change the velocities $\dot{x}_1, \dot{x}_2$ rapidly. Exemplary for this is the bouncing ball in contact with a solid surface and only one equation of motion [5]. A force impulse given to the ball in the course of the collision is exhausted in this case by the gravity directed to the opposite during the following jumping phase.

5 Conclusion

The model description by means of delta functions permits the inclusion of point abstractions, also called sharp concentrated quantities, in the model equations. Alternatively, a piecewise description is possible by using fitting conditions that connect the different solution intervals. Both description forms are equivalent. The closed representation by delta functions, however, is closer to the problem because the point abstractions also appear in the differential equations. In case of piecewise description they do not appear in the model equations but additional fitting conditions characterize their influence [5]. Internally the integration takes place in both cases between time and state events, respectively. The piecewise modelling in simulation coming from the classical way of looking should be completed therefore by using delta functions which permit a closed form of the model description.

References

[1] ACSL, Reference Manual, Version 11, Mitchel & Gauthier Associates [MGA] Inc., Concord MA, USA, 1995.

[2] L. Collatz, *Differentialgleichungen*, Teubner Studienbücher Mathematik, B.G. Teubner, Stuttgart, 1990.

[3] G. Hauger, W. Schnell, D. Gross, *Technische Mechanik, Band 3: Kinetik*, vierte Auflage (Springer-Lehrbuch), Springer-Verlag, 1993.

[4] J. Heinhold and U. Kulisch, *Analogrechnen, Eine Einführung*, BI-Hochschultaschenbücher, Bibliographisches Institut AG, Mannheim, 1969.

[5] R. Hohmann, *Methoden und Modelle der Kontinuierlichen Simulation*, (Berichte aus der Informatik), Shaker Verlag, Aachen, 1999, Zugl.: Habil.-Schrift, Universität Magdeburg, 1998.

[6] R. Hohmann, *Delta- und Thetafunction in Continuous Models*, in Proceedings of the 1994 Summer Computer Simulation Conference, San Diego, California, 1994, pp. 775–779.

[7] W. Macke, *Elektromagnetische Felder, Ein Lehrbuch der Theoretischen Physik*, Akademische Verlagsanstalt Geest & Portig K.-G., Leipzig, 1960.