MULTIBODY DYNAMICS 2009, ECCOMAS Thematic Conference
K. Arczewski, J. Frączek, M. Wojtyra (eds.)
Warsaw, Poland, 29 June – 2 July 2009

SUCCESSIVE PROJECTION METHOD FOR THE SIMULATION OF SPATIAL DYNAMICS OF MULTIBODIES

Dmitry Vlasenko*, Roland Kasper *

* Institute of Mobile Systems (IMS), Otto-von-Guericke-University Magdeburg
Universitätsplatz 2, D-39016 Magdeburg, Germany
e-mail: Dmitri.Vlasenko@ovgu.de, Roland.Kasper@ovgu.de ,
web page: http://imat.mb.uni-magdeburg.de/

Keywords: Projection, Dynamics, Multibody, Euler parameters, component-oriented simulation.

Abstract. *The successive coordinate projection method efficiently stabilizes mechanical constraints when the when the non-minimal number of orientation coordinates is used. The implementation of successive approach for standard stabilization methods and for distributed stabilization methods significantly reduces the numerical cost of simulation. The proposed algorithms of successive stabilizations were tested with a Yamaha YZF-R1 motorcycle engine model and with a KUKA KR 15/2 industrial manipulator model. The simulation results show that the successive coordinate projection is stable and can be implemented for complex mechanical systems.*

1 INTRODUCTION

Absolute coordinates are widely used in general-purpose multibody simulation software for the description of motion of mechanical systems. In this case the configuration of a rigid body is defined by the global position vector of the origin of the body coordinate system and by a set of orientation coordinates, describing the orientation of the body coordinate system with respect to the global coordinate system.

The minimal set of orientation coordinates includes three independent parameters (e.g. Euler angles, Bryan angles, Rodriguez parameters, etc.) [15, 21]. The main drawback of this representation of rotation is that all three-parameter systems have singular positions in those locations where these parameters are not defined unequivocally. The proofs that no representation of finite rotations by three parameters is possible without singular points can be found in [16] and in [10].

In order to avoid the singularity problem, non-minimal sets of orientation coordinates are widely used (e.g. Euler parameters, etc.). But the use of non-minimal sets implies that the additional dependencies of orientation coordinates should be satisfied during the simulation of motion of a multibody system. Let *n* be the number of bodies in a mechanical system, $\mathbf{x}^i = \left(x_1^i, x_2^i, x_3^i\right)^T$ be the vector of position coordinates of body *i* and $\mathbf{e}^i$ be the vector of orientation coordinates of body *i*. Then the vector of absolute coordinates of the *i*-th body $\mathbf{q}^i$ can be written as

$$\mathbf{q}^i = \begin{pmatrix} \mathbf{x}^i \\ \mathbf{e}^i \end{pmatrix} \tag{1}$$

By $\mathbf{q} = (\mathbf{q}^{1T} \quad ... \quad \mathbf{q}^{nT})^T$ denote the vector of absolute coordinates of a multibody system. The equation of additional constraints for the simulated system can be written in the following form

$$\mathbf{g}_b(\mathbf{q}) = \mathbf{0} \tag{2}$$

For example, for Euler parameters we have: $\mathbf{e}^i = (e_0^i \quad e_1^i \quad e_2^i \quad e_3^i)^T$ and $\mathbf{g}_b = (\mathrm{g}_\mathrm{b}^1, \quad ... \quad \mathrm{g}_\mathrm{b}^n)^T$, where

$$\mathrm{g}_b^i(\mathbf{q}^i) = \sum_{k=0}^{3} \left(e_k^i\right)^2 - 1 \qquad i = 1..n \tag{3}$$

The use of non-minimal set implies also the dependencies of the first time derivative of coordinates

$$\frac{\mathrm{d}}{\mathrm{dt}} \mathbf{g}_b(\mathbf{q}) = \mathbf{G}_b(\mathbf{q}) \cdot \dot{\mathbf{q}} = \mathbf{0} \tag{4}$$

where $\mathbf{G}_b$ is the Jacobian of $\mathbf{g}_b$

$$\mathbf{G}_b(\mathbf{q}) = \frac{\partial \mathbf{g}_b(\mathbf{q})}{\partial \mathbf{q}} \tag{5}$$

Clear, that for Euler parameters example we have

$$\mathbf{G}_b(\mathbf{q}) = \mathrm{diag}\left(2\mathbf{s}^1, \ldots, 2\mathbf{s}^n\right) \tag{6}$$

where $\mathbf{s}^i = (0 \quad 0 \quad 0 \quad e_0^i \quad e_1^i \quad e_2^i \quad e_3^i)$.

Let $\mathbf{v}^i$ be the vector of generalized velocities of the *i*-th body:

$$\mathbf{v}^i=\begin{pmatrix}\dot{\mathbf{x}}^i\\ \boldsymbol{\omega}^i\end{pmatrix} \tag{7}$$

where $\boldsymbol{\omega}^i$ is the angular velocity of the *i*-th body. By $\mathbf{T}^i$ denote the relation matrix between the vector of the first time derivative of coordinates $\dot{\mathbf{q}}^i$ of the *i*-th body and the vector of bodies generalized velocities $\mathbf{v}^i$:

$$\dot{\mathbf{q}}^i=\mathbf{T}^i(\mathbf{q}^i)\mathbf{v}^i \tag{8}$$

For Euler parameters example we have the following formula for the calculation of $\mathbf{T}^i$ [13]:

$$\mathbf{T}^i(\mathbf{q}^i)=\begin{pmatrix}\mathbf{I_3} & \mathbf{0}\\ \mathbf{0} & \frac{1}{2}\mathbf{E}^i(\mathbf{e}^i)\end{pmatrix} \tag{9}$$

where $\mathbf{I_3}$ is the (3,3) identity matrix and

$$\mathbf{E}^i(\mathbf{e}^i)=\begin{pmatrix}-e_1^i & -e_2^i & -e_3^i\\ e_0^i & e_3^i & -e_2^i\\ -e_3^i & e_0^i & e_1^i\\ e_2^i & -e_1^i & e_0^i\end{pmatrix} \tag{10}$$

Let $\mathbf{T}$ be the relation matrix between the first time derivative of the absolute coordinates $\dot{\mathbf{q}}$ and the velocities $\mathbf{v}$:

$$\dot{\mathbf{q}}=\mathbf{T}(\mathbf{q})\mathbf{v} \tag{11}$$

From the definition follows that $\mathbf{T}=\mathrm{diag}(\mathbf{T}^1,\dots,\mathbf{T}^n)$. Obviously, if we use the non-minimal set of coordinates, then $\mathbf{T}$ has more rows than columns.

Substituting (11) in (4), we get

$$\mathbf{G}_b(\mathbf{q})\cdot\dot{\mathbf{q}}=\mathbf{G}_b(\mathbf{q})\cdot\mathbf{T}(\mathbf{q})\cdot\mathbf{v}=\mathbf{0} \tag{12}$$

This equation should be fulfilled for arbitrary $\mathbf{v}$. Therefore, we obtain the following important relation between $\mathbf{G}_b$ and $\mathbf{T}$: for each $\mathbf{q}$, satisfying (2), we have

$$\mathbf{G}_b(\mathbf{q})\cdot\mathbf{T}(\mathbf{q})=\mathbf{0} \tag{13}$$

Let $\mathbf{g}(\mathbf{q})$ denote the vector of constraints, describing joints, connecting bodies in the simulated mechanical system:

$$\mathbf{g}(\mathbf{q})=\mathbf{0} \tag{14}$$

By $\mathbf{G}(\mathbf{q})$ denote the Jacobian matrix of $\mathbf{g}(\mathbf{q})$:

$$\mathbf{G}(\mathbf{q})=\frac{\partial\mathbf{g}(\mathbf{q})}{\partial\mathbf{q}} \tag{15}$$

Differentiating (14), we get the equations of joint constraints on the velocity level:

$$\mathbf{g}_\mathbf{v}(\mathbf{q},\mathbf{v})=\frac{\mathrm{d}}{\mathrm{dt}}\mathbf{g}(\mathbf{q})=\mathbf{G}(\mathbf{q})\cdot\dot{\mathbf{q}}=\mathbf{G}(\mathbf{q})\cdot\mathbf{T}(\mathbf{q})\cdot\mathbf{v}=\mathbf{0} \tag{16}$$

Differentiating (16) one more time, we obtain the equation of constraints on the acceleration level

$$\mathbf{g}_{\mathbf{vv}}(\mathbf{q},\mathbf{v},\dot{\mathbf{v}})=\mathbf{G}(\mathbf{q})\cdot\mathbf{T}(\mathbf{q})\cdot\dot{\mathbf{v}}-\mathbf{u}(\mathbf{q},\mathbf{v})=\mathbf{0} \tag{17}$$

where $\mathbf{u}(\mathbf{q},\mathbf{v})$ is a vector that absorbs terms that are quadratic in the velocities

$$\mathbf{u}(\mathbf{q},\mathbf{v}) = \frac{\partial(\mathbf{G}(\mathbf{q})\cdot\dot{\mathbf{q}})}{\partial\mathbf{q}}\cdot\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q})\frac{\partial(\mathbf{T}(\mathbf{q})\cdot\mathbf{v})}{\partial\mathbf{q}}\cdot\dot{\mathbf{q}} \tag{18}$$

The equations of constraints can be simplified, if we define the matrix $\widehat{\mathbf{G}}$ as a product

$$\widehat{\mathbf{G}} = \mathbf{G}\cdot\mathbf{T} \tag{19}$$

Then (16) can be written as

$$\mathbf{g}_{\mathbf{v}}(\mathbf{q},\mathbf{v}) = \widehat{\mathbf{G}}(\mathbf{q})\cdot\mathbf{v} = \mathbf{0} \tag{20}$$

Substituting (19) in (17), we get:

$$\widehat{\mathbf{G}}(\mathbf{q})\cdot\dot{\mathbf{v}} = \mathbf{u}(\mathbf{q},\mathbf{v}) \tag{21}$$

Combining (21) with Newton-Euler equations of motion, we obtain the index-one formulation of the equations of motion [8]

$$\dot{\mathbf{q}} = \mathbf{T}(\mathbf{q})\mathbf{v} \tag{22}$$

$$\begin{pmatrix} \mathbf{M}(\mathbf{q}) & \widehat{\mathbf{G}}(\mathbf{q}) \\ \widehat{\mathbf{G}}(\mathbf{q}) & \mathbf{0} \end{pmatrix} \cdot \begin{pmatrix} \dot{\mathbf{v}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{q},\mathbf{v}) \\ \mathbf{u}(\mathbf{q},\mathbf{v}) \end{pmatrix} \tag{23}$$

where $\mathbf{f}(\mathbf{q},\mathbf{v})$ is the vector of external forces, $\mathbf{M}$ is the mass matrix, $\boldsymbol{\lambda}$ is the vector of Lagrange multipliers.

After solving (23) for the accelerations $\dot{\mathbf{v}}$, the values of the coordinates $\overline{\mathbf{q}}$ and the velocities $\overline{\mathbf{v}}$ are calculated, using standard ODE integration schemes (e.g. Runge-Kutta or a multistep method). But the coordinates $\overline{\mathbf{q}}$ do not fulfil the equations of joint constraints (14) nor the additional constraint equations (2). The equations of constraints on the velocity level (20) are also not satisfied. The drift of the constraints on the coordinate and on the velocity level grows with time t – at worst quadratically [2]. To overcome this difficulty, the coordinates and velocities can be projected back onto the manifolds given by the constraints on the position and on the velocity level.

In this article we show how to improve the numerical efficiency of the coordinate stabilization by implementation of the successive projection of coordinates on the manifold, given by of additional constraint, and on the manifold, given by joint constraints.

We propose also to implement the successive approach for the method of distributed stabilization, used in the component-oriented simulation of multibodies. The successive version of distributed stabilization needs much less numerical operations than the non-succesive one.

The proposed methods were implemented for the simulation of dynamics of two complex mechanical models: Yamaha YZF-R1 motorcycle engine and KUKA KR 15/2 industrial manipulator.

2 STANDARD COORDINATE PROJECTION METHOD

Usually in the description of the coordinate projection method is assumed that the equations of motion (22), (23) are written for the vectors of transformed velocities $\check{\mathbf{v}}=\dot{\mathbf{q}}$ and transformed accelerations $\dot{\check{\mathbf{v}}} = \ddot{\mathbf{q}}$ [5, 14, 4, 15]. This formulation has a significant disadvantage that the size of equations of motion for $\dot{\check{\mathbf{v}}}$ is larger than the size of (23), therefore, its solution needs more numerical operations.

Let us consider now the projection of coordinates when the original equations of motion (22), (23) are used. In the standard projection method the constraints $\mathbf{g}_{\boldsymbol{b}}$ and $\mathbf{g}$ are stabilized simultaneously using the projection of coordinates on the manifold, given by the $\mathbf{g}_{\boldsymbol{b}}$ and $\mathbf{g}$

$$\mathbf{g_A} = \begin{pmatrix} \mathbf{g} \\ \mathbf{g_b} \end{pmatrix} = \mathbf{0} \tag{24}$$

Let $\mathbf{G_A}$ denote the Jacobian of $\mathbf{g_A}$

$$\mathbf{G_A} = \frac{\partial \mathbf{g_A}}{\partial \mathbf{q}} = \begin{pmatrix} \mathbf{G} \\ \mathbf{G_b} \end{pmatrix} \tag{25}$$

Then the projection of $\overline{\mathbf{q}}$ onto the manifold given by $\mathbf{g_A}$ can be calculated as [4, 14]

$$\mathbf{q} = \overline{\mathbf{q}} + \mathbf{\Delta_A} \tag{26}$$

where the stabilizing displacement $\mathbf{\Delta_A}$ is the minimal norm solution of the equation

$$\mathbf{G_A}(\overline{\mathbf{q}}) \cdot \mathbf{\Delta_A} = -\mathbf{g_A}(\overline{\mathbf{q}}) \tag{27}$$

If all rows of $\mathbf{G_A}$ are independent, then $\mathbf{\Delta_A}$ can be calculated using the following formula

$$\mathbf{\Delta_A} = -\mathbf{G_A^T}(\mathbf{G_A}\mathbf{G_A^T})^{-1}\mathbf{g_A} \tag{28}$$

Otherwise, in the presence of redundant constraints we need to perform the singular value decomposition of $\mathbf{G_A} \in R^{m \times s}$ [Golub]:

$$\mathbf{G_A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{29}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, the matrix $\mathbf{\Sigma}$ a (m,s) diagonal matrix with nonnegative numbers on the diagonal: $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0) \in R^{m \times s}$. Then $\mathbf{\Delta_A}$ can be calculated as

$$\mathbf{\Delta_A} = -\mathbf{V}\mathbf{\Sigma}^{+}\mathbf{U}^T\mathbf{g_A} \tag{30}$$

where $\mathbf{\Sigma}^{+} = \mathrm{diag}(\frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_r}, 0, \ldots, 0) \in R^{m \times s}$.

If necessary, we can repeat the projection procedure several times, in order to fulfill the equation of constraints [1, 3, 4]. In order to reduce the calculation cost, the decomposition of the same matrix $\mathbf{G_A}(\overline{\mathbf{q}})$ can be implemented on each projection step, i.e. the stabilizing displacement $\mathbf{\Delta_A^{(i)}}$ on the i-th projection step can be found as a minimal norm solution of the equation:

$$\mathbf{G_A}(\overline{\mathbf{q}}) \cdot \Delta_A^{(i)} = -\mathbf{g_A}\left(\overline{\mathbf{q}}^{(i-1)}\right) \tag{31}$$

where $\overline{\mathbf{q}}^{(i-1)}$ is the projected value of coordinates, calculated after the previous projection step.

Clear, that there is no additional constraints between velocities, therefore, on the velocity level we need only to stabilize (20), using the following formula

$$\mathbf{v} = \overline{\mathbf{v}} + \zeta \tag{32}$$

where ζ is the minimal norm solution of the equation

$$\widehat{\mathbf{G}}(\overline{\mathbf{q}}) \cdot \zeta = -\mathbf{g_v}(\mathbf{q}) \tag{33}$$

This approach has several disadvantages. Firstly, the decomposition of $\mathbf{G_A}(\overline{\mathbf{q}})$, performed during the stabilization of coordinates, cannot be used for the stabilization of velocities because on the velocity level we need the decomposition of $\widehat{\mathbf{G}}(\overline{\mathbf{q}})$. Secondly, the size of $\mathbf{G_A}(\overline{\mathbf{q}})$ is large than the size of $\widehat{\mathbf{G}}(\overline{\mathbf{q}})$. Despite of the sparse structure of $\mathbf{G_A}(\overline{\mathbf{q}})$, its decomposition needs more numerical operations than the decomposition of $\widehat{\mathbf{G}}(\overline{\mathbf{q}})$ and of $\mathbf{G_b}(\mathbf{q})$.

3 SUCCESSIVE STABILIZATION METHOD

3.1 Main idea of the method

In order to avoid the disadvantages of the standard projection method we propose to split the process of stabilization 0of additional and joint constraints.

Firstly, we calculate from the vector of coordinates $\overline{\mathbf{q}}$ its projection $\overline{\overline{\mathbf{q}}}$ onto the manifold given by additional constraints $\mathbf{g}_b(\overline{\mathbf{q}})$. Then we calculate the value of $\mathbf{q}$ as the projection of $\overline{\overline{\mathbf{q}}}$ on the manifold given by joint constraints $\mathbf{g}(\overline{\overline{\mathbf{q}}})$ in such way that $\mathbf{q}$ satisfy also the additional constraints, i.e. $\mathbf{g}_b(\mathbf{q})\approx\mathbf{0}$.

3.2 Step one: Stabilization of $\mathbf{g}_b$

The projection of $\overline{\mathbf{q}}$ onto the manifold given by $\mathbf{g}_b$ is similar to the projection onto $\mathbf{g}_\mathbf{A}$:

$$\overline{\overline{\mathbf{q}}} = \overline{\mathbf{q}} + \Delta_b \tag{34}$$

where the stabilizing displacement $\boldsymbol{\Delta}_\mathbf{b}$ is the minimal norm solution of the equation:

$$\mathbf{G}_b(\overline{\mathbf{q}})\cdot\boldsymbol{\Delta}_b = -\mathbf{g}_b(\overline{\mathbf{q}}) \tag{35}$$

For the Euler parameters example $\mathbf{g}_b(\overline{\mathbf{q}})$ and $\mathbf{G}_b$ are calculated from (3), (6). Substituting $\mathbf{G}_b$ from (6) in (28), we get: $\boldsymbol{\Delta}_b = (\boldsymbol{\Delta}_b^{1T}, \quad ... \quad \boldsymbol{\Delta}_b^{nT})^T$ where:

$$\boldsymbol{\Delta}_b^i = -\frac{1}{2}\overline{\mathbf{q}}^i \frac{g_b^i(\overline{\mathbf{q}}^i)}{g_b^i(\overline{\mathbf{q}}^i)+1} \quad i = 1..n \tag{36}$$

Now we need to prove that the first step does not increase significantly the order of the mistake of the joint constraints $\mathbf{g}(\overline{\mathbf{q}})$. Let us assume that the matrix $\mathbf{G}_b$ has a full row rank. Then, implementing (27) for the calculation of the minimal norm solution, we get

$$\boldsymbol{\Delta}_\mathbf{b} = -\mathbf{G}_b^T(\mathbf{G}_\mathbf{b}\mathbf{G}_b^T)^{-1}\mathbf{g}_\mathbf{b} \tag{37}$$

From (27) follows that

$$\mathbf{G}_\mathbf{b}(\overline{\mathbf{q}})\cdot\boldsymbol{\Delta}_\mathbf{A} = -\mathbf{g}_\mathbf{b}(\overline{\mathbf{q}}) \tag{38}$$

$$\mathbf{G}(\overline{\mathbf{q}})\cdot\boldsymbol{\Delta}_\mathbf{A} = -\mathbf{g}(\overline{\mathbf{q}}) \tag{39}$$

Therefore, the mistake of the joint constraints before the stabilization of additional constraints can be estimated as

$$\|\mathbf{g}(\overline{\mathbf{q}})\| = \|\mathbf{G}(\overline{\mathbf{q}})\cdot\boldsymbol{\Delta}_\mathbf{A}\| = O(\|\boldsymbol{\Delta}_A\|) \tag{40}$$

Using (39), (37), we get

$$\begin{aligned}\|\mathbf{g}(\overline{\mathbf{q}} + \boldsymbol{\Delta}_\mathbf{b})\| \approx \|\mathbf{g}(\overline{\mathbf{q}}) + \mathbf{G}(\overline{\mathbf{q}})\cdot\boldsymbol{\Delta}_\mathbf{b}\| = \|-\mathbf{G}\cdot\boldsymbol{\Delta}_\mathbf{A} - \mathbf{G}\cdot\mathbf{G}_b^T(\mathbf{G}_\mathbf{b}\mathbf{G}_b^T)^{-1}\mathbf{g}_\mathbf{b}\| = \\ = \|\mathbf{G}\cdot[\mathbf{I}-\mathbf{G}_b^T(\mathbf{G}_\mathbf{b}\mathbf{G}_b^T)^{-1}\mathbf{G}_\mathbf{b}]\cdot\boldsymbol{\Delta}_\mathbf{A}\| = O(\|\boldsymbol{\Delta}_A\|)\end{aligned} \tag{41}$$

Therefore, the order of the mistake of joint constraints did not change during the first step of stabilization.

3.3 Step two: Stabilization of g

On the second step we project $\overline{\overline{\mathbf{q}}}$ onto the manifold given by $\mathbf{g}$

$$\mathbf{q} = \overline{\overline{\mathbf{q}}} + \boldsymbol{\Delta} \tag{42}$$

where $\mathbf{q}$ is the vector of projected coordinates and $\mathbf{\Delta}$ is the stabilizing displacement. We propose to calculate $\mathbf{\Delta}$ proportional to the matrix $\mathbf{T}$

$$\mathbf{\Delta} = \mathbf{T}(\bar{\bar{\mathbf{q}}})\cdot\boldsymbol{\delta} \tag{43}$$

where $\boldsymbol{\delta}$ is some increment. Therefore, using the standard formula for the calculation of stabilizing displacement, we obtain that $\boldsymbol{\delta}$ can be calculated as the minimal norm solutions of the equation

$$\mathbf{G}(\bar{\bar{\mathbf{q}}})\cdot\mathbf{T}(\bar{\bar{\mathbf{q}}})\cdot\boldsymbol{\delta} = -\mathbf{g}(\bar{\bar{\mathbf{q}}}) \tag{44}$$

Or, in another form

$$\widehat{\mathbf{G}}(\bar{\bar{\mathbf{q}}})\cdot\boldsymbol{\delta} = -\mathbf{g}(\bar{\bar{\mathbf{q}}}) \tag{45}$$

Now we need to prove that projected coordinates $\mathbf{q}$ satisfy also the additional constraints $\mathbf{g_b}$. Using (35), we get

$$\mathbf{g_b}(\mathbf{q}) = \mathbf{g_b}(\bar{\bar{\mathbf{q}}} + \Delta) \approx \mathbf{g_b}(\bar{\bar{\mathbf{q}}}) + \mathbf{G_b}(\bar{\bar{\mathbf{q}}})\cdot\Delta = \mathbf{G_b}(\bar{\bar{\mathbf{q}}})\cdot\mathbf{T}(\bar{\bar{\mathbf{q}}})\cdot\boldsymbol{\delta} \tag{46}$$

From (13) follows that $\mathbf{G_b}(\bar{\bar{\mathbf{q}}})\cdot\mathbf{T}(\bar{\bar{\mathbf{q}}}) \approx \mathbf{0}$, since $\mathbf{g}_b(\bar{\bar{\mathbf{q}}}) \approx \mathbf{0}$. Thus, from (46) follows that

$$\mathbf{g_b}(\mathbf{q}) \approx \mathbf{0} \tag{47}$$

4 IMPLEMENTATION OF THE SUCCESSIVE STABILIZATION METHOD

In order to decrease the drift of the model, we propose to repeat the projection of coordinates two times after each time step, using the same decomposition of the matrix $\widehat{\mathbf{G}}\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)$. This can be written as

$$\begin{aligned}
\bar{\bar{\mathbf{q}}}^{(1)} &= \bar{\mathbf{q}} + \mathbf{\Delta}_b^{(1)} \\
\mathbf{q}^{(1)} &= \bar{\bar{\mathbf{q}}}^{(1)} + \mathbf{T}\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)\cdot\boldsymbol{\delta}^{(1)} \\
\bar{\bar{\mathbf{q}}}^{(2)} &= \mathbf{q}^{(1)} + \mathbf{\Delta}_b^{(2)} \\
\mathbf{q} &= \bar{\bar{\mathbf{q}}}^{(2)} + \mathbf{T}\left(\bar{\bar{\mathbf{q}}}^{(2)}\right)\cdot\boldsymbol{\delta}^{(2)}
\end{aligned} \tag{48}$$

where $\boldsymbol{\delta}^{(1)}$, $\boldsymbol{\delta}^{(2)}$ are the minimal norm solutions of equations

$$\widehat{\mathbf{G}}\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)\cdot\boldsymbol{\delta}^{(i)} = -\mathbf{g}\left(\bar{\bar{\mathbf{q}}}^{(i)}\right) \quad i=1,2 \tag{49}$$

and $\mathbf{\Delta}_b^{(1)}$, $\mathbf{\Delta}_b^{(2)}$ are the minimal norm solutions of the following equations

$$\begin{aligned}
\mathbf{G}_b(\bar{\mathbf{q}})\cdot\mathbf{\Delta}_b^{(1)} &= -\mathbf{g}_b(\bar{\mathbf{q}}) \\
\mathbf{G}_b\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)\cdot\mathbf{\Delta}_b^{(2)} &= -\mathbf{g}_b\left(\mathbf{q}^{(1)}\right)
\end{aligned} \tag{50}$$

The projection of velocities can be performed in a similar way

$$\begin{aligned}
\mathbf{v}^{(1)} &= \bar{\mathbf{v}} + \zeta^{(1)} \\
\mathbf{v} &= \mathbf{v}^{(1)} + \zeta^{(2)}
\end{aligned} \tag{51}$$

and $\zeta^{(1)}$, $\zeta^{(2)}$ are the minimal norm solutions of the following equations

$$\begin{aligned} \widehat{\mathbf{G}}\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)\cdot\boldsymbol{\zeta}^{(1)} &= -\mathbf{g}_{\mathbf{v}}(\mathbf{q},\bar{\mathbf{v}}) \\ \widehat{\mathbf{G}}\left(\bar{\bar{\mathbf{q}}}^{(1)}\right)\cdot\boldsymbol{\zeta}^{(2)} &= -\mathbf{g}_{\mathbf{v}}\left(\mathbf{q},\mathbf{v}^{(1)}\right) \end{aligned} \tag{52}$$

5 EXAMPLE 1: YAMAHA YZF-R1 ENGINE

The proposed successive projection method of was implemented in object-oriented simulation software **V**irtual **S**ystems **D**esigner (VSD) [12, 18-20]. The integration of VSD with a CAD tool Autodesk Inventor allows simulating the dynamics of 3D-CAD models without additional redesign.

Fig. 1 shows an Autodesk Inventor model of Yamaha YZF-R1 Motorcycle Engine.

Figure 1: CAD model of Yamaha YZF-R1 Motorcycle Engine

In the description of the model all units are of the SI (Système International: kg, m, s, Pa) unless otherwise indicated. The most important parameters of the engine are shown in Table 1 [22].

Engine type	Liquid-cooled, 4-stroke, DOHC
Displacement	$9.98\cdot10^{-4}$ (998 cm^3)
Cylinder arrangement	Forward-inclined parallel 4-cylinder
Bore *b*	0.074
Stroke	0.058
Compression ratio *r*	11.8
Engine idling speed	104.7 ~ 115.2 (1000 ~ 1100 rpm)

Table 1 Parameters of Yamaha YZF-R1 Motorcycle Engine

The correspondent mechanical system consists of 10 bodies (4 pistons, 4 connecting rods, the crankshaft and the engine block, fixed in the space), connected by 13 joints (4 cylindrical joints, connecting rods and pistons, 4 cylindrical joints, connecting pistons and the engine block, 4 revolute joints, connecting the crankshaft and rods, and 1 revolute joint, connecting

the crankshaft and the engine block). In order to emulate the influence of the flywheel on the motion of the engine, we significantly increased the moment of inertia of the crankshaft. We simulated the dynamics of the engine during the engine cranking, adding the following force and torques:

1. Starter motor torque, acting on the crankshaft during the first five revolutions

$$\tau = \begin{cases} 22 & t < t_1 \\ 11 & t_1 < t < t_2 \end{cases} \tag{53}$$

where t_1 denote the time when the crankshaft reaches the speed 41.89 rad/s (400 rpm), t_2 denote the time when the crankshaft makes five revolutions.

2. Gas pressure force, acting on the piston throughout the Otto-cycle

$$f_z = (p - p_a) \cdot b^2 \cdot \frac{\pi}{4} \tag{54}$$

where b is the engine bore, shown in Table 1, p_a is the atmospheric pressure and p is the gas pressure in the combustion chamber, calculated as

$$p = \begin{cases} p_a & intake\ stroke \\ p_a \cdot (v/v_{max})^{-\gamma} - p_a & compression\ stroke \\ p_i \cdot (v/v_{min})^{-\gamma} - p_a & expansion\ stroke \end{cases} \tag{55}$$

Here p_i is the ignition pressure, v is the volume of the combustion chamber, v_{max} is the maximal volume of the combustion chamber, v_{min} is the minimal volume of the combustion chamber. The ratio between v_{max} and v_{min} is equal to the compression ratio r, shown in Table 1.

Fig. 2 shows the Pressure-Volume diagram, correspondent to the engine idling, when $p_i = 50p_a$.

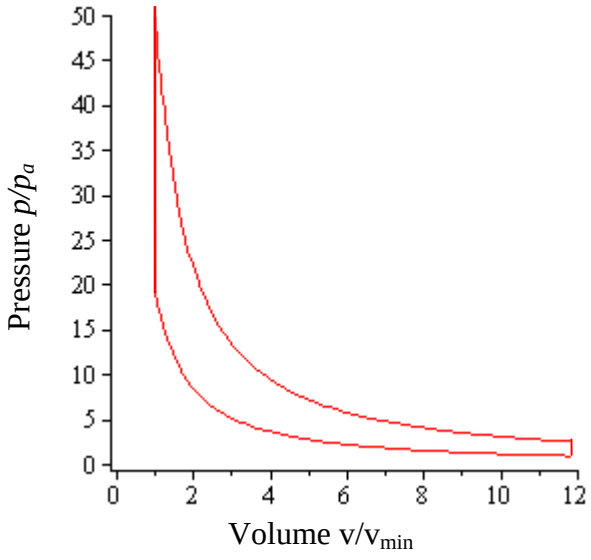

Figure 2: Pressure-Volume diagram of Otto cycle

3. Torque, acting on the crankshaft, which emulates the friction action [6]

$$\tau = 4 \cdot \pi \cdot \mathrm{v_d} \cdot p_t \tag{56}$$

where $\mathrm{v_d} = 4 \cdot (\mathrm{v_{max}} - \mathrm{v_{min}})$ is the engine displacement volume and p_t is the total motored friction, calculated as [9]

$$p_t = 10^5 \left(0.97 + 0.15 \frac{v_\mathrm{N}}{1000} + 0.05 \frac{v_\mathrm{N}^2}{1000} \right) \tag{57}$$

Here v_N is the angular velocity of the crankshaft, measured in revolutions per minute.

In Fig. 3 is shown the angular velocity of the crankshaft, measured in revolutions per minute in the cases when the gas pressure forces act in all four cylinders or in two cylinders or only in one cylinder. Obviously, the reduction of number of working cylinders significantly increases the irregularity in the crankshaft velocity.

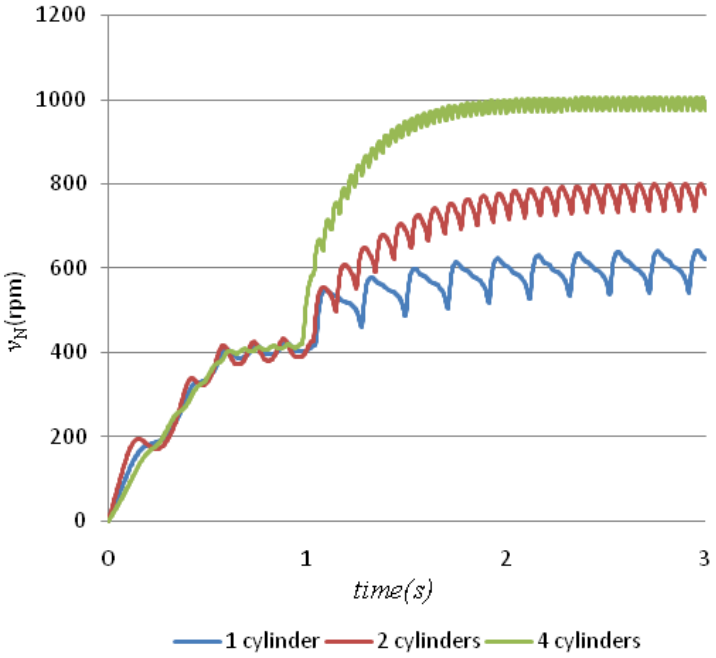

Figure 3: Angular velocity of the crankshaft

We simulated the dynamics of the model with Runge-Kutta method of the second order with the fixed time step equal to 0.002 *s*. In Fig. 4 is shown the drift on the coordinate level $\|\mathbf{g}_\mathrm{A}\|$ and the drift on the velocity level $\|\mathbf{g}_\mathbf{v}\|$. The simulation data shows that the algorithm is stable and the model's drift is constant.

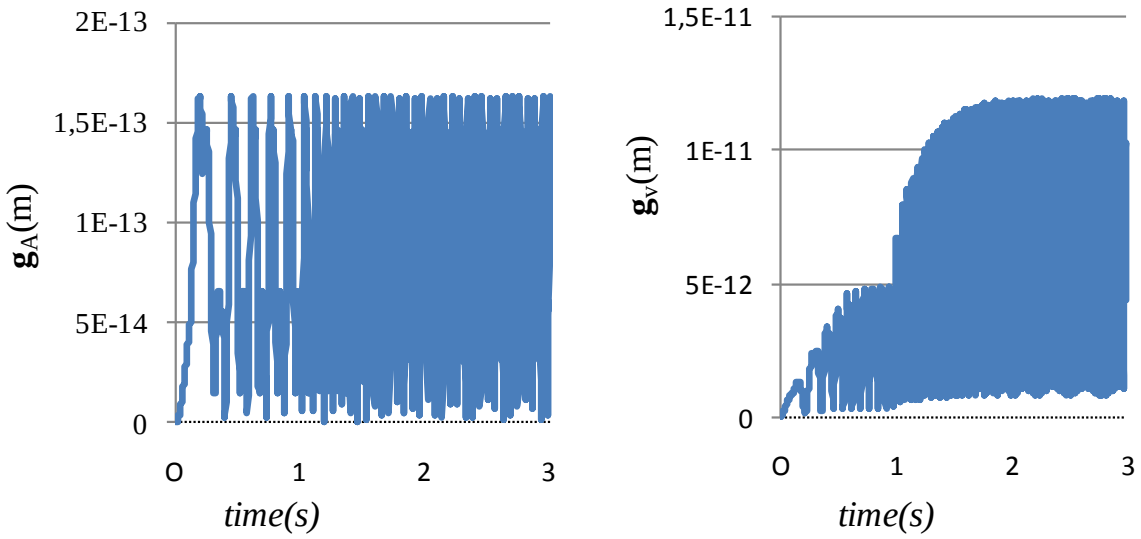

Figure 4: Drift of the model a) on the coordinate level b) on the velocity level

6 DISTRIBUTED SUCCESSIVE COORDINATE PROJECTION METHOD FOR THE COMPONENT-ORIENTED SIMULATION OF MULTIBODIES

In the last years we developed and implemented a method, based on the projection algorithm for absolute coordinates, which performs the component-oriented simulation of multibodies [17, 18]. In our method the model's partition, defined during the model's specification, remains during the simulation, i.e. we use the simulation based on the hierarchy of subsystems.

The main advantages of the simulation on the basis of subsystems are:

- Each subsystem can be modeled, tested and compiled independently. This significantly decreases the time and cost of the models' development and test.
- The commercial classified information of submodels is protected. A submodel works like a "black box" that has to provide only the strictly determined set of information via its interfaces. All submodel's internal data: parameters of constraints, forces, masses of internal bodies, etc. are unknown to the users of submodels.
- Critical effects like Coulomb friction, backslash etc. can be encapsulated inside a subsystem.
- Subsystems are ideal candidates for the partitioning of systems on multiple processors.
- Mechanical subsystems are represented by separate objects which interact via predefined interfaces with each other. Using such interface, simulation model can be easily extended by electronic and control components.

Fig. 5 shows the object-oriented method of the simulation of mechanical systems, implemented in VSD [6]. The base idea of the method is to perform the simulation of mechanical systems using the hierarchy of submodels that builds up the complete system.

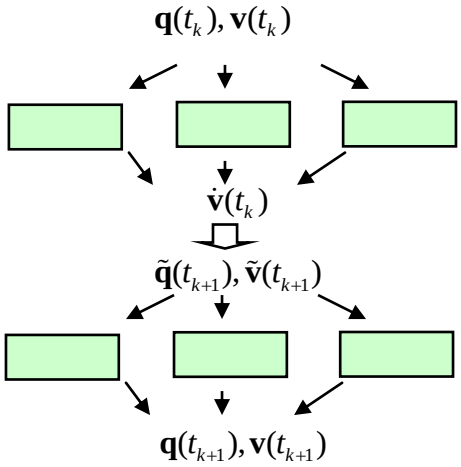

Figure 5: Data flow in simulation steps

Submodels of the first level in general consist of connected bodies. Submodels of next levels consist, without loss of generality, of connected submodels. Since the main number of calculations proceeds inside of submodels, it follows that the simulation can be distributed easily on several processors. During the simulation at each time step the following tasks have to be performed:

1. Distributed calculation of the absolute accelerations $\dot{\mathbf{v}}(t_k)$.
2. Calculation of the absolute coordinates and velocities at the next time step. Using a favorite ODE integration scheme (e.g. Runge-Kutta or some multistep method), the value of the absolute coordinates $\tilde{\mathbf{q}}(t_{k+1})$ and velocities $\tilde{\mathbf{v}}(t_{k+1})$ at the new time step can be obtained.
3. Distributed stabilization of the absolute coordinates $\mathbf{q}(t_{k+1})$ and velocities $\mathbf{v}(t_{k+1})$.

The detailed description of the distributed component-oriented stabilization can be found in [18]. The significant disadvantage of this method is that the stabilization of subsystem's constraints on the coordinate level needs the decomposition of the constraint Jacobian matrix $\mathbf{G(q)}$ and the stabilization on the velocity level needs the decomposition of $\widehat{\mathbf{G}}= \mathbf{G}\cdot\mathbf{T}$.

We successfully implemented the successive approach for the algorithm of distributed stabilization in a similar way as described above for the non-distributed stabilization. The successive stabilization need much less numerical operations because now the decomposition of the same matrix $\widehat{\mathbf{G}}$ is used both for the stabilization of coordinates and velocities.

7 EXAMPLE 2: INDUSTRIAL MANIPULATOR KUKA KR 15/2

The successive distributed projection method was implemented in object-oriented simulation software VSD. In order to test our software for the simulation of dynamics of realistic CAD models we developed an Autodesk Inventor model of the industrial manipulator KUKA KR 15/2 [11]. This is a six-axis robot with articulated kinematics for all continuous-path controlled tasks. The main areas of application of KR 15/2 are handling, assembly, machining, etc.

The complete Autodesk Inventor model consists of 1036 parts coupled in several subsystems, shown in Fig 6: upper arm, motors, cyclo-drive gearboxes, etc. The correspondent VSD model includes 43 bodies connected by 95 joints. Some of model constraints are redundant because of the model's design in Autodesk Inventor (e.g. the definition of stiff connection as three plane-to-plane joints leads to the generation of three redundant constraints).

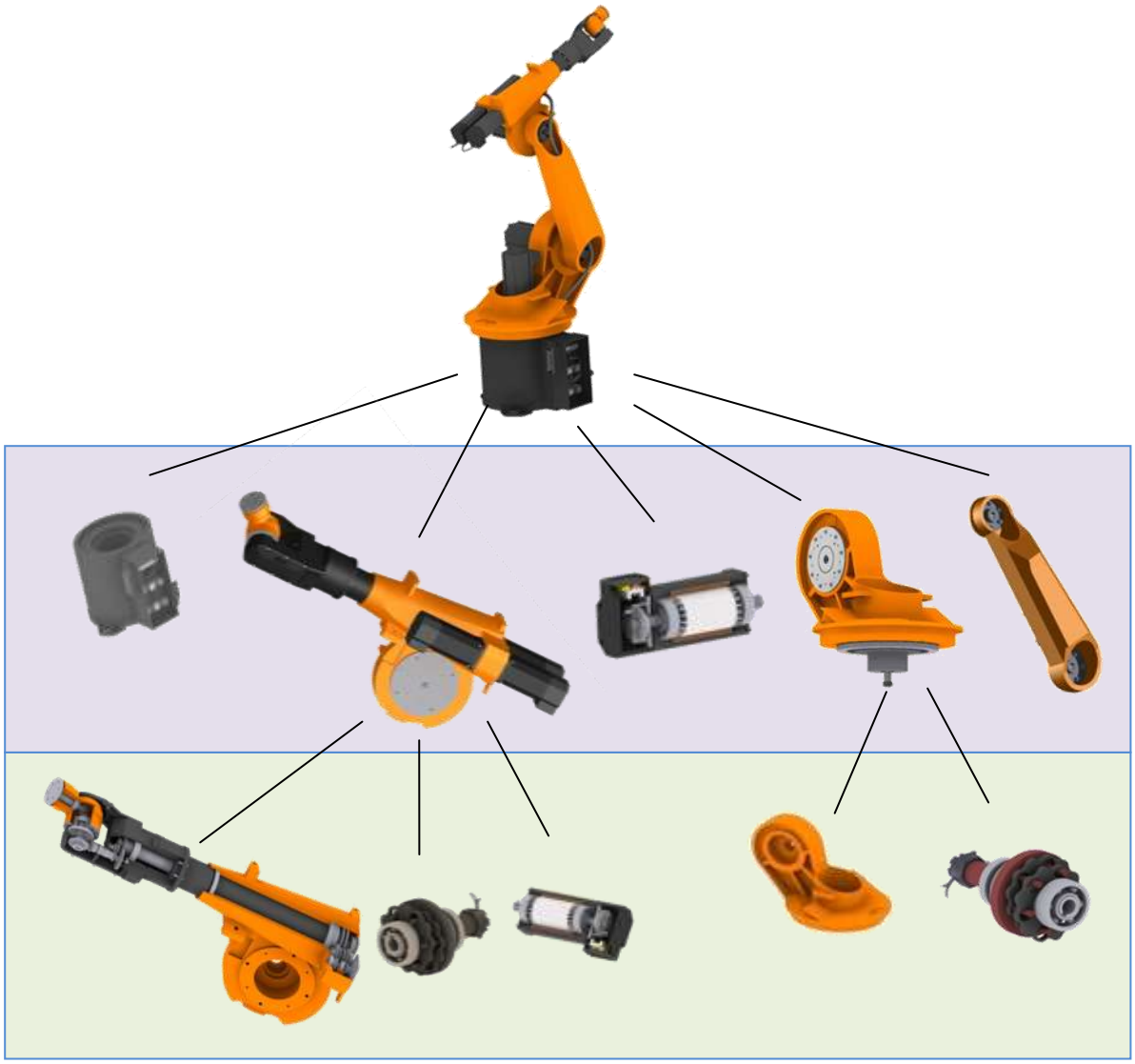

Figure 6: Hierarchy of subsystems of KR 15/2 model

The simulation data shows that the consequent distributed stabilization algorithm is stable and the model's drift is constant. The implementation of the successive approach for the distributed stabilization greatly reduces numerical costs.

8 CONCLUSION

In the case, when the non-minimal number of orientation coordinates is used for the description of bodies' configuration, the implementation of standard stabilization method leads to the increased size of equations of motion. This disadvantage can be avoided, if the projection of coordinates is performed successively on the manifold given by additional constraints and on the manifold given by joint constraints.

The successive stabilization approach can be also implemented for the component-oriented simulation of multibodies. The successive version of distributed stabilization needs much less numerical operations than the non- successive one.

The algorithms were tested with models of Yamaha YZF-R1 motorcycle engine and of KUKA KR 15/2 industrial manipulator. The simulation results show that the successive coordinate projection is stable, numerically efficient and can be implemented for complex mechanical systems with redundant constraints.

REFERENCES

[1] U. Ascher, H. Chin, L. Petzold, S. Reich. Stabilization of constrained mechanical systems with DAEs and invariant manifolds. *Mech. Struct. & Mach*. 23, 135-157, 1995.

[2] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Comp. Meth. in Appl. Mech. and Eng.*, 1:1–16, 1972

[3] H. Chin. *Stabilization Methods for Simulations of Constrained Multibody Dynamics*. Ph.D. Thesis, University of British Columbia, Vancouver, Canada (1995).

[4] E. Eich-Soellner, C. Führer: *Numerical Methods in Multibody Dynamics*, B. G. Teubner, Stuttgart, (1998.

[5] E. Eich. *Projizierende Mehrschrittverfahren zur numerischen Losung der Bewegungsgleichungen technischer Mehrkorpersysteme mit Zwangsbedingungen und Unstetigkeiten.* PhD thesis, Institut für Mathematik, Universität Augsburg, 1991. ”VDI-Fortschrittsberichte”, Reihe 18, Nr. 109,VDI-Verlag, Düsseldorf, 1992.

[6] Colin R. Ferguson. *Internal Combustion Engines*: Applied Thermosciences, 1st edn., Wiley, New York, 1986.

[7] Gene H. Golub, Charles F. van Loan. *Matrix Computations*, 3rd ed., Johns Hopkins UP, 1996.

[8] E.J. Haug. *Computer Aided Kinematics and Dynamics of Mechanical Systems Volume I: Basic Methods*. Allyn & Bacon, Boston, 1989.

[9] J. B. Heywood. *Internal Combustion Engine Fundamentals*, McGraw-Hill, Inc., New York, 1988.

[10] H. Hopf. Systeme symmetrischer Bilinearformen und Euklidische Modelle der projektiven Räume. *Naturf. Ges.*, Zürich, 165–177, 1940.

[11] Spezifikation Roboter KR 6/2, KR 15/2, KR 15 L6/2
http://www.kuka.com/NR/rdonlyres/B6067DF9-EA5D-4037-9080-77E7C4F59D0D/0/spez_kr6_de_en_fr.pdf

[12] R. Kasper, D. Vlasenko, G. Sintotskiy. A Component Oriented Approach to Multidisciplinary Simulation of Mechatronic Systems. Proceedings of the EUROSIM Congress on Modelling and Simulation (EUROSIM 2007), Ljubljana, Slovenia, September 9-13, 2007.

[13] A. L. Schwab and J. P. Meijaard. How to draw Euler angles and utilize Euler parameters, In Proceedings of IDETC/CIE 2006, ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Philadelphia, PA, CD-ROM, ASME, New York, September 10-13, 2006.

[14] Reinhold von Schwerin. *Multibody System Simulation. Numerical Methods, Algorithms and Software*. Springer, 1999.

[15] A. A. Shabana. *Computational Dynamics*, John Wiley and Sons, 2001.

[16] J. Stuelpnagel. On the parameterization of the three-dimensional rotation group. *Siam Rev.* 6, No 4:422–430, 1964.

[17] D. Vlasenko**,** R. Kasper. Algorithm for Component Based Simulation of Multibody Dynamics. *Technische Mechanik*, Band 26, Heft 2, 2006, pp. 92-105, 2006.

[18] D. Vlasenko, R. Kasper. Integration Method of CAD Systems. Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2007, Las Vegas, Nevada, USA, September 4-7, 2007.

[19] D. Vlasenko, R. Kasper. A New Software Approach for the Simulation of Multibody Dynamics. *ASME Journal of Computational and Nonlinear Dynamics*, Volume 2, Issue 3, 2007, pp. 274-278, 2007.

[20] D. Vlasenko, R. Kasper. Implementation of the Symbolic Simplification for the Calculation of Accelerations of Multibodies. Proceedings of Industrial Simulation Conference 2008, Lyon, France, 9-11 June, 2008.

[21] Jens Wittenburg. *Dynamics of Multibody Systems*. Springer-Verlag Berlin Heidelberg 2008.

[22] YZF-R1P/YZF-R1PC SERVICE MANUAL, 2001 by Yamaha Motor Corporation, U.S.A., First edition