KYBERNETIKA — VOLUME *49* (2013), NUMBER 5, PAGES 765–779

EFFICIENT APPLICATION OF E–INVARIANTS IN FINITE ELEMENT METHOD FOR AN ELASTODYNAMIC EQUATION

Martin Balazovjech and Ladislav Halada

We introduce a new efficient way of computation of partial differential equations using a hybrid method composed from FEM in space and FDM in time domain. The overall computational scheme is explicit in time. The key idea of the suggested way is based on a transformation of standard basis functions into new basis functions. The results of this matrix transformation are e-invariants (effective invariants) with such suitable properties which save the number of arithmetical operations needed for a problem solution. The application of this procedure and its effectiveness for 2D problem was the first time published in [2]. Now we describe the generalization of this procedure for 3D problem. In order to present the main principle of our process and its advantage, we first explain the main idea of our approach on a simple 1D example and then the application of the e-invariants on an elastodynamics equation using hexahedral elements in 3D is described. Finally, the efficiency of the suggested method in both cases from the point of the required number of arithmetical operations is analyzed. The result of this analysis confirms computational efficiency the suggested method and the usefulness of e-invariants which save only the essential information needed for the computation. Moreover, the method can be used for various types of elements and equations.

Keywords: e–invariants, finite element method, elastodynamics equation, efficient computation

Classification: 65M60

1. INTRODUCTION

Computation of an elastodynamic equation occurs in material science, seismology and many other applications. Due to complicated geometry of the boundary domain and heterogeneity of the area, computation of the elastodynamic equation is usually realistic enough only if the computation domain is very large. Especially in seismological problems parallel computing on supercomputers has to be used. Therefore, we must economize storage during the computation. There are suggestions in the literature how this problem can be overcome. For example, the global mass matrix can be approximated by a diagonal matrix (lumped mass matrix) [5, 7, 9, 10]. The lumped mass matrix saves memory and offers faster computation. Another possibility is using the global restoring force vector beside of the global stiffness matrix in the problem formulation. It means, we do not assemble the global stiffness matrix by local matrices but we

assemble the global restoring forces vector using the local restoring forces. Such method saves computer memory considerably, but on the other hand it makes the computation more time consuming, because the restoring force vector is computed in every time level without saving the stiffness matrix in memory [1, 2, 6, 8]. As we present in this paper, this drawback can be eliminated by our approach: using the new basis functions and the e-invariants which are defined in the subsequent sections. As it will be proved, the efficiency of our approach is due to the fact that the e-invariants contain only the minimum necessary information for the local restoring force vector computation. This cannot be said for the standard approach.

Concerning the computational speed-up, our approach was used in the seismology laboratory of Comenius University in Bratislava and in the parallel data laboratory at Carnegie Mellon University in Pittsburgh. A detailed analysis of computational efficiency improvement using our method is explained in [11], where the Chino Hills earthquake 2008 was simulated, and the speed-up about 3 is reported.

The paper is organized as follows. Section 2 establishes the problem formulation and the basic idea of using new basis functions on simple 1D example. Section 3 introduces the application of the e-invariants to the motion equation for 3D case using the problem formulation with the local restoring force vector. The role of the e-invariants for 3D case is described in Section 4. The number of arithmetical operations required for computations by this method is presented for cases.

2. THE PROBLEM FORMULATION AND THE BASIC IDEA OF USING THE NEW BASIS FUNCTIONS

The numerical formulation of the elastodynamics equation of motion using FEM in space has generally the following matrix form

$$\mathsf{M}\mathsf{u}_{,tt} + \mathsf{K}\mathsf{u} = \mathsf{f} \tag{1}$$

where M and K are the global mass and stiffness matrices, respectively, f is the global force term. Discrete solution of this equation is usually based on the central difference time approximation of the second order for the second time derivative. Thus, after this approximation and the rearrangement of the variable vectors $\mathbf{u}$ according to time order, we obtain the explicit recurrent formula

$$\mathsf{u}^{m+1} = \Delta t^2 \mathsf{M}^{-1}(\mathsf{f}^m - \mathsf{K}\mathsf{u}^m) - \mathsf{u}^{m-1} + 2\mathsf{u}^m. \tag{2}$$

In practical applications the order of M and K can be quite large $(10^6 - 10^8)$ [8]. Especially, in such cases attention has to be paid to formulation of such a computational algorithm that economize the storage and the number of required computational operations. However, there are also situations, when the global mass and stiffness matrices are too large and hence they can not be stored in the computer memory and a numerical solution of the original problem can not be computed by finite element method. Therefore, efforts to overcome this problem can be found in literature. For example, the global mass matrix can be approximated by a diagonal matrix (lumped mass matrix) and beside using the global stiffness matrix some authors suggest to use global restoring force vector $r^m = \mathsf{K}\mathsf{u}^m$. It means that the stiffness matrix is not computed in order to

reduce the storage memory, but rather the restoring force vector r^m is computed in every time level without saving the stiffness matrix in the memory. In other words we do not assemble the global stiffness matrix using the local matrices but we assemble the global restoring forces using the local restoring forces. This procedure is described for example in [1, 6]. This way one can achieve important reduction of storage requirements. Moreover, direct computation of the restoring forces can be also used in the case of nonlinear material response [4].

As result, this computational process reduces the storage, but is very time consuming. In order to eliminate this problem we have used transformation which plays a key role in different research areas.

In the following section we suggest how to formulate the computation of the restoring force vector to be the most effective from the computational point of view.

First of all, we will analyze our approach on the simple 1D problem represented by the following equation

$$\rho u_{x,tt} = \tau_{xx,x} + f_x. \tag{3}$$

Let the variables x and u_x be interpolated by the linear functions

$$\mathbf{b} = [b_1, b_2]^T = \frac{1}{2}[(1-\xi), (1+\xi)]^T \tag{4}$$

in the form

$$x = [b_1, b_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{b}^T \mathbf{x} \qquad u_x = [b_1, b_2] \begin{bmatrix} u_{x1} \\ u_{x2} \end{bmatrix} = \mathbf{b}^T \mathbf{u}_x. \tag{5}$$

Then equation (3) can be expressed after discretization in the displacement weak form on the element $\langle x_1, x_2 \rangle$ as follows

$$\int_{-1}^{1} \left[\mathbf{b}\mathbf{b}^T \rho \mathbf{u}_{x,tt} + (\lambda + 2\mu)\, \mathbf{b}_{,x} \mathbf{b}_{,x}^T\, \mathbf{u}_x - \mathbf{b} f_x \right] x_{,\xi}\, \mathrm{d}\xi = 0 \tag{6}$$

and thus, the stiffness matrix and the restoring force vector for this problem have the form

$$\begin{aligned} \mathbf{K} &= \int_{-1}^{1} \left[(\lambda + 2\mu)(\mathbf{b}_{,x}\, \mathbf{b}_{,x}^T) x_{,\xi} \right] \mathrm{d}\xi = \frac{\lambda + 2\mu}{x_2 - x_1} \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} \\ \mathbf{r}_x &= \mathbf{K}\mathbf{u}_x = \frac{\lambda + 2\mu}{x_2 - x_1} \begin{bmatrix} u_{x1} - u_{x2} \\ -u_{x1} + u_{x2} \end{bmatrix} \end{aligned} \tag{7}$$

where λ and μ are Lame elastic coefficients. The mass matrix can be approximated by the lumped matrix. This mathematical formulation corresponds with the problem depicted in Figure 1. This is a standard way how one can compute the matrix $\mathbf{K}$ and corresponding vector $\mathbf{r}_x$ by the values $\mathbf{x}^T = [x_1, x_2]$ and $\mathbf{u}_x^T = [u_{x1}, u_{x2}]$. But what is noteworthy, these coordinates of the string position and the nodal displacements keep more information than is needed for the restoring force vector computation. The values $[x_1, x_2]$ keep the information where the spring is located and $[u_{x1}, u_{x2}]$ where the nodal values will be shifted under the local forces. But the local restoring force vector is

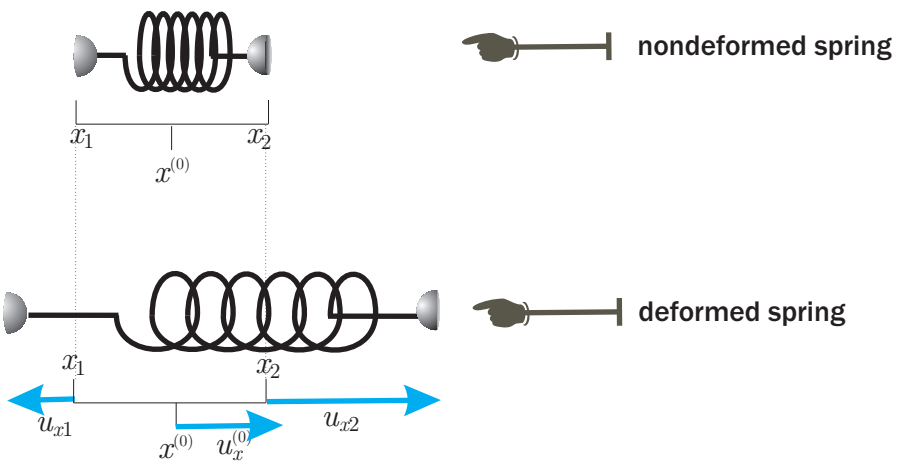

Fig. 1. Nodal values x_1, x_2 determine the spring position on the line at rest and u_{x1}, u_{x2} are displacements at the nodes under a local strength.

independent on this information. What is needed for its computation is the length and the elongation of the spring, i. e. the values $(x_2 - x_1)$ and $(-u_{x1} + u_{x2})$ (see Figure 1). Moreover, on the basis of the action-reaction law, $r_{x1} = -r_{x2}$ holds and therefore only one vector component of $\mathbf{r}_x$ is satisfactory to be computed.

Therefore, beside of the vector $\mathbf{x}$, $\mathbf{u}_x$ and $\mathbf{r}_x$ we define the following new types of vectors

1. the vector of a segment transformation on the interval (x_1, x_2)
   $\mathbf{x}^{(inv)} = [x^{(0)}, x^{(1)}]^T = \frac{1}{2}[(x_1 + x_2), (x_2 - x_1)]^T$
2. the vector of a displacement
   $\mathbf{u}_x^{(inv)} = [u_x^{(0)}, u_x^{(1)}]^T = \frac{1}{2}[(u_{x1} + u_{x2}), (u_{x2} - u_{x1})]^T$
3. the vector of a local restoring forces
   $\mathbf{r}_x^{(inv)} = [r_x^{(0)}, r_x^{(1)}]^T = \frac{1}{2}[(r_{x1} + r_{x2}), (r_{x2} - r_{x1})]^T$

and the matrix for 1D transformation

$$\mathbf{T}_1 = \begin{bmatrix} +1 & +1 \\ -1 & +1 \end{bmatrix} \tag{8}$$

with the property

$$\begin{gathered} \tfrac{1}{2}\mathbf{T}_1\mathbf{T}_1^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{I}_1 \\ \mathbf{b}^{(inv)} = \mathbf{T}_1\mathbf{b} \quad \mathbf{x}^{(inv)} = \tfrac{1}{2}\mathbf{T}_1\mathbf{x} \quad \mathbf{u}_x^{(inv)} = \tfrac{1}{2}\mathbf{T}_1\mathbf{u}_x \quad \mathbf{r}_x^{(inv)} = \tfrac{1}{2}\mathbf{T}_1\mathbf{r}_x \end{gathered} \tag{9}$$

where components $x^{(1)}$ is invariant with respect to the location of element, $u_x^{(1)}$ to the translation of element and $r_x^{(1)}$ to the global force acting on the element, respectively. As a result of this computational process we obtain

$$\mathbf{r}_x = \mathbf{T}_1^T \mathbf{r}_x^{(inv)} = \begin{bmatrix} r_{x1} \\ r_{x2} \end{bmatrix} \tag{10}$$

for the local restoring force vector. The interpretation of the coordinates of the vectors $\mathbf{x}^{(inv)}$ and $\mathbf{u}_x^{(inv)}$ is obvious. The first coordinates $x^{(0)}$ and $u_x^{(0)}$ represent the position the of the spring and the place where the spring has been shifted, respectively. These values are not needed for the restoring force computation. However, second coordinates are needed because the value $x^{(1)}$ represents the length of the spring and $u_x^{(1)}$ the change of its length (see Figure 1). These invariants have also another very useful property. The standard FEM computational algorithm requires to know the derivative of x and u_x with respect to the local variable ξ. It can be verified easily that in the case of the standard base

$$x_{,\xi} = -\frac{1}{2}x_1 + \frac{1}{2}x_2 \qquad u_{x,\xi} = -\frac{1}{2}u_{x1} + \frac{1}{2}u_{x2} \tag{11}$$

holds, and in the new base $\mathbf{b}^{(inv)} = [1, \xi]$

$$x_{,\xi} = 0x^{(0)} + 1x^{(1)} = x^{(1)} \qquad u_{x,\xi} = 0u_x^{(0)} + 1u_x^{(1)} = u_x^{(1)} \tag{12}$$

holds. Thus we can see advantage of using the e-invariant version for the restoring force computation. Although this advantage is insignificant for 1D problem, we show in the next sections that this approach is expedient for 3D case.

3. APPLICATION OF THE E-INVARIANTS TO THE EQUATION OF MOTION — 3D CASE

The wave propagation in a three-dimensional elastic body satisfies the equation of motion

$$\begin{aligned} \rho u_{x,tt} &= \tau_{xx,x} + \tau_{xy,y} + \tau_{xz,z} + f_x \\ \rho u_{y,tt} &= \tau_{yx,x} + \tau_{yy,y} + \tau_{yz,z} + f_y \\ \rho u_{z,tt} &= \tau_{zx,x} + \tau_{zy,y} + \tau_{zz,z} + f_z \end{aligned} \tag{13}$$

and Hooke's law [5, 9]

$$\begin{bmatrix} \tau_{xx} \\ \tau_{yy} \\ \tau_{zz} \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{xz} \end{bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \begin{bmatrix} u_{x,x} \\ u_{y,y} \\ u_{z,z} \\ u_{x,y} + u_{y,x} \\ u_{y,z} + u_{z,y} \\ u_{x,z} + u_{z,x} \end{bmatrix} \tag{14}$$

where ρ is mass density, u_i and f_i are components of a displacement and body force per unit volume, respectively, τ_{ij} are components of the stress tensor in $i, j = x, y, z$ direction. The computational domain is $\bar{\Omega} = \Omega \cup \Gamma$, where Ω represents the interior

and Γ is the border of the domain. In the sequel, we will consider again the weak form approximation of equations (13) over any quadrilateral element Ω^e

$$\begin{aligned}
&\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}(\mathbf{b}\rho u_{x},_{tt}+\mathbf{b},_{x}\,\tau_{xx}+\mathbf{b},_{y}\,\tau_{xy}+\mathbf{b},_{z}\,\tau_{xz}-\mathbf{b}f_x)\,Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta\\
&\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}(\mathbf{b}\rho u_{y},_{tt}+\mathbf{b},_{x}\,\tau_{yx}+\mathbf{b},_{y}\,\tau_{yy}+\mathbf{b},_{z}\,\tau_{yz}-\mathbf{b}f_y)\,Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta\\
&\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}(\mathbf{b}\rho u_{z},_{tt}+\mathbf{b},_{x}\,\tau_{zx}+\mathbf{b},_{y}\,\tau_{zy}+\mathbf{b},_{z}\,\tau_{zz}-\mathbf{b}f_z)\,Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta
\end{aligned} \tag{15}$$

where the components of vector $\mathbf{b} = [b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8]^T$ are Lagrange family interpolation functions $b_i = \frac{1}{8}(1+\xi_i\xi)(1+\eta_i\eta)(1+\zeta_i\zeta)$, $i = 1, 2, 3, \cdots, 8$ defined on the master cube element given by the coordinates

$$\begin{aligned}
\boldsymbol{\xi} &= [-1,+1,-1,+1,-1,+1,-1,+1]^T\\
\boldsymbol{\eta} &= [-1,-1,+1,+1,-1,-1,+1,+1]^T\\
\boldsymbol{\zeta} &= [-1,-1,-1,-1,+1,+1,+1,+1]^T
\end{aligned} \tag{16}$$

and $Det\mathbf{J}$ is determinant of the Jacobian transformation. If the actual element Ω^e is given by the coordinates

$$\begin{aligned}
\mathbf{x} &= [\,x_1,\,x_2,\,x_3,\,x_4,\,x_5,\,x_6,\,x_7,\,x_8\,]^T\\
\mathbf{y} &= [\,y_1,\,y_2,\,y_3,\,y_4,\,y_5,\,y_6,\,y_7,\,y_8\,]^T\\
\mathbf{z} &= [\,z_1,\,z_2,\,z_3,\,z_4,\,z_5,\,z_6,\,z_7,\,z_8\,]^T
\end{aligned} \tag{17}$$

and the displacements by

$$\begin{aligned}
\mathbf{u}_x &= [u_{x1},\,u_{x2},\,u_{x3},\,u_{x4},\,u_{x5},\,u_{x6},\,u_{x7},\,u_{x8}]^T\\
\mathbf{u}_y &= [u_{y1},\,u_{y2},\,u_{y3},\,u_{y4},\,u_{y5},\,u_{y6},\,u_{y7},\,u_{y8}]^T\\
\mathbf{u}_z &= [u_{z1},\,u_{z2},\,u_{z3},\,u_{z4},\,u_{z5},\,u_{z6},\,u_{z7},\,u_{z8}]^T
\end{aligned} \tag{18}$$

then the transformation between the master element and the actual element, as well as the approximation of the dependent variables u_x, u_y and u_z on the master element Ω^e can be expressed in the well known isoparametric form

$$\begin{aligned}
x &= \mathbf{b}^T\mathbf{x} \qquad u_x = \mathbf{b}^T\mathbf{u}_x\\
y &= \mathbf{b}^T\mathbf{y} \qquad u_y = \mathbf{b}^T\mathbf{u}_y\\
z &= \mathbf{b}^T\mathbf{z} \qquad u_z = \mathbf{b}^T\mathbf{u}_z.
\end{aligned} \tag{19}$$

Applying the relation (19) to the integral equations (15) we obtain the matrix form equation for the element Ω^e

$$\begin{bmatrix}\mathbf{M} & 0 & 0\\ 0 & \mathbf{M} & 0\\ 0 & 0 & \mathbf{M}\end{bmatrix}\begin{bmatrix}\mathbf{u}_x,_{tt}\\ \mathbf{u}_y,_{tt}\\ \mathbf{u}_z,_{tt}\end{bmatrix}+\begin{bmatrix}\mathbf{r}_x\\ \mathbf{r}_y\\ \mathbf{r}_z\end{bmatrix}=\begin{bmatrix}\mathbf{f}_x\\ \mathbf{f}_y\\ \mathbf{f}_z\end{bmatrix} \tag{20}$$

where the local restoring force vector for this element is composed of the following components

$$\begin{aligned}
\mathbf{r}_x &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} (\mathbf{b}\mathbf{b}_{,x}\,\tau_{xx} + \mathbf{b}_{,y}\,\tau_{xy} + \mathbf{b}_{,z}\,\tau_{xz})\, Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \\
\mathbf{r}_y &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} (\mathbf{b}\mathbf{b}_{,x}\,\tau_{yx} + \mathbf{b}_{,y}\,\tau_{yy} + \mathbf{b}_{,z}\,\tau_{yz})\, Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \\
\mathbf{r}_z &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} (\mathbf{b}\mathbf{b}_{,x}\,\tau_{zx} + \mathbf{b}_{,y}\,\tau_{zy} + \mathbf{b}_{,z}\,\tau_{zz})\, Det\mathbf{J}\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta
\end{aligned} \tag{21}$$

while the numerical integration can be calculated by Gauss quadrature. Naturally, equation (20) holds only for a hexahedral element of the domain. We remember that for the computation of the vectors $\mathbf{r}_x, \mathbf{r}_y$ and $\mathbf{r}_z$ we need to know the vectors

$$\begin{aligned}
\mathbf{b}_{,x} &= \mathbf{b}_{,\xi}\,\xi_{,x} + \mathbf{b}_{,\eta}\,\eta_{,x} + \mathbf{b}_{,\zeta}\,\zeta_{,x} \\
\mathbf{b}_{,y} &= \mathbf{b}_{,\xi}\,\xi_{,y} + \mathbf{b}_{,\eta}\,\eta_{,y} + \mathbf{b}_{,\zeta}\,\zeta_{,y} \\
\mathbf{b}_{,z} &= \mathbf{b}_{,\xi}\,\xi_{,z} + \mathbf{b}_{,\eta}\,\eta_{,z} + \mathbf{b}_{,\zeta}\,\zeta_{,z}
\end{aligned} \tag{22}$$

where

$$\begin{bmatrix} \xi_{,x} & \eta_{,x} & \zeta_{,x} \\ \xi_{,y} & \eta_{,y} & \zeta_{,y} \\ \xi_{,z} & \eta_{,z} & \zeta_{,z} \end{bmatrix} = \frac{1}{Det\mathbf{J}} \begin{bmatrix} y_{,\eta}\,z_{,\zeta} - y_{,\zeta}\,z_{,\eta} & y_{,\zeta}\,z_{,\xi} - y_{,\xi}\,z_{,\zeta} & y_{,\xi}\,z_{,\eta} - y_{,\eta}\,z_{,\xi} \\ x_{,\zeta}\,z_{,\eta} - x_{,\eta}\,z_{,\zeta} & x_{,\xi}\,z_{,\zeta} - x_{,\zeta}\,z_{,\xi} & x_{,\eta}\,y_{,\xi} - x_{,\xi}\,z_{,\eta} \\ x_{,\eta}\,y_{,\zeta} - x_{,\zeta}\,y_{,\eta} & x_{,\zeta}\,y_{,\xi} - x_{,\xi}\,y_{,\zeta} & x_{,\xi}\,y_{,\eta} - x_{,\eta}\,y_{,\xi} \end{bmatrix} \tag{23}$$

while

$$\mathbf{J} = \begin{bmatrix} x_{,\xi} & x_{,\eta} & x_{,\zeta} \\ y_{,\xi} & y_{,\eta} & y_{,\zeta} \\ z_{,\xi} & z_{,\eta} & z_{,\zeta} \end{bmatrix} \tag{24}$$

and

$$\begin{aligned}
b_{i,\xi} &= \tfrac{1}{8}\xi_i\,(1+\eta_i\eta)\,(1+\zeta_i\zeta) \\
b_{i,\eta} &= \tfrac{1}{8}\eta_i\,(1+\xi_i\xi)\,(1+\zeta_i\zeta) \\
b_{i,\zeta} &= \tfrac{1}{8}\zeta_i\,(1+\xi_i\xi)\,(1+\eta_i\eta)
\end{aligned} \tag{25}$$

are components of $\mathbf{b}_{,\xi}\,, \mathbf{b}_{,\eta}\,\mathbf{b}_{,\zeta}$.

To solve the problem on the whole domain, we have to make so called assembling of all elements in the domain with the aim to construct the system of equations for the whole domain.

Now, as in 1D case we express the restoring force vector by the e-invariants in 3D space. Let $\mathbf{b}^{(inv)}$ be a new vector of basis functions on the unit square and $\mathbf{T}_3$ be a corresponding transform matrix between $\mathbf{b}$ and $\mathbf{b}^{(inv)}$. Let us define the vector $\mathbf{b}^{(inv)}$ and matrix $\mathbf{T}_3$ as follows

$$\mathbf{b}^{(inv)} = \left[1,\ \xi,\ \eta,\ \eta\xi,\ \zeta,\ \xi\zeta,\ \eta\zeta,\ \eta\xi\zeta\right]^T \tag{26}$$

and

$$\mathbf{T}_3 = \begin{bmatrix} +\mathbf{T_2} & +\mathbf{T_2} \\ -\mathbf{T_2} & +\mathbf{T_2} \end{bmatrix} \quad \mathbf{T_2} = \begin{bmatrix} +\mathbf{T_1} & +\mathbf{T_1} \\ -\mathbf{T_1} & +\mathbf{T_1} \end{bmatrix} \quad \mathbf{T_1} = \begin{bmatrix} +1 & +1 \\ -1 & +1 \end{bmatrix} \tag{27}$$

with the property

$$\tfrac{1}{8}\mathbf{T}_3\mathbf{T}_3^T = \mathbf{I}_3 \quad \Rightarrow \mathbf{T}_3^{-1} = \tfrac{1}{8}\mathbf{T}_3^T \quad \text{and} \quad \mathbf{b}^{(inv)} = \mathbf{T}_3\mathbf{b} \quad \Rightarrow \mathbf{b} = \tfrac{1}{8}\mathbf{T}_3^T\mathbf{b}^{(inv)} \tag{28}$$

where $\mathbf{I}_3$ is the unit square matrix with dimension 8. These properties are very important because, as we can see, they again facilitate to express the value of the function in nodal points using the new basis functions. Really, the following identity

$$x = \mathbf{b}^T\mathbf{x} = \mathbf{b}^T\mathbf{I}_3\mathbf{x} = \mathbf{b}^T\left(\tfrac{1}{8}\mathbf{T}_3^T\mathbf{T}_3\right)\mathbf{x} = \left(\mathbf{b}^T\mathbf{T}_3^T\right)\left(\tfrac{1}{8}\mathbf{T}_3\mathbf{x}\right) = \mathbf{b}^{(inv)T}\mathbf{x}^{(inv)} \tag{29}$$

holds. By the same way we obtain

$$\begin{array}{lll} & y = \mathbf{b}^{(inv)T}\mathbf{y}^{(inv)} & z = \mathbf{b}^{(inv)T}\mathbf{y}^{(inv)} \\ u_x = \mathbf{b}^{(inv)T}\mathbf{u}_x^{(inv)} & u_y = \mathbf{b}^{(inv)T}\mathbf{u}_y^{(inv)} & u_z = \mathbf{b}^{(inv)T}\mathbf{u}_z^{(inv)} \end{array} \tag{30}$$

what can be expressed in the following matrix form

$$\begin{bmatrix} x \\ y \\ z \\ u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} x^{(0)} & x^{(1)} & x^{(2)} & x^{(12)} & x^{(3)} & x^{(13)} & x^{(23)} & x^{(123)} \\ y^{(0)} & y^{(1)} & y^{(2)} & y^{(12)} & y^{(3)} & y^{(13)} & y^{(23)} & y^{(123)} \\ z^{(0)} & z^{(1)} & z^{(2)} & z^{(12)} & z^{(3)} & z^{(13)} & z^{(23)} & z^{(123)} \\ u_x^{(0)} & u_x^{(1)} & u_x^{(2)} & u_x^{(12)} & u_x^{(3)} & u_x^{(13)} & u_x^{(23)} & u_x^{(123)} \\ u_y^{(0)} & u_y^{(1)} & u_y^{(2)} & u_y^{(12)} & u_y^{(3)} & u_y^{(13)} & u_y^{(23)} & y^{(123)} \\ u_z^{(0)} & u_z^{(1)} & u_z^{(2)} & u_z^{(12)} & u_z^{(3)} & u_z^{(13)} & u_z^{(23)} & u_z^{(123)} \end{bmatrix} \begin{bmatrix} 1 \\ \xi \\ \eta \\ \eta\xi \\ \zeta \\ \xi\zeta \\ \eta\zeta \\ \eta\xi\zeta \end{bmatrix} \tag{31}$$

where the components $x^{(k)}, y^{(k)}, z^{(k)}, u_x^{(k)}, u_y^{(k)}, u_z^{(k)}, \quad k = 0, 1, 2, 12, 3, 13, 23, 123$ of the matrix in expression 31 are *e-invariants* with suitable properties from the computational point of view and the vectors $(x^{(k)}, y^{(k)})$ are the plane invariants and their interpretation is described in the next section. It is easy to see from (30) that we can derive all needed relations between the values defined in the standard basis space and the values defined in the new basis space. For example, the derivative of the displacement fulfills

$$\begin{array}{lllllllll} u_{x,x} & = & \mathbf{b}_{,x}^{(inv)T}\,\mathbf{u}_x^{(inv)} & u_{x,y} & = & \mathbf{b}_{,y}^{(inv)T}\,\mathbf{u}_x^{(inv)} & u_{x,z} & = & \mathbf{b}_{,z}^{(inv)T}\,\mathbf{u}_x^{(inv)} \\ u_{y,x} & = & \mathbf{b}_{,x}^{(inv)T}\,\mathbf{u}_y^{(inv)} & u_{y,y} & = & \mathbf{b}_{,y}^{(inv)T}\,\mathbf{u}_y^{(inv)} & u_{y,z} & = & \mathbf{b}_{,z}^{(inv)T}\,\mathbf{u}_y^{(inv)} \\ u_{z,x} & = & \mathbf{b}_{,x}^{(inv)T}\,\mathbf{u}_z^{(inv)} & u_{z,y} & = & \mathbf{b}_{,y}^{(inv)T}\,\mathbf{u}_z^{(inv)} & u_{z,z} & = & \mathbf{b}_{,z}^{(inv)T}\,\mathbf{u}_z^{(inv)} \end{array} \tag{32}$$

and the relation between the derivative of $\mathbf{b}$ and $\mathbf{b}^{(inv)T}$ follows from (28)

$$\begin{array}{lll} \mathbf{b}_{,x} = & \mathbf{T}_3^{-1}\mathbf{b}_{,x}^{(inv)} & = \tfrac{1}{8}\mathbf{T}_3^T\mathbf{b}_{,x}^{(inv)} \\ \mathbf{b}_{,y} = & \mathbf{T}_3^{-1}\mathbf{b}_{,y}^{(inv)} & = \tfrac{1}{8}\mathbf{T}_3^T\mathbf{b}_{,y}^{(inv)} \\ \mathbf{b}_{,z} = & \mathbf{T}_3^{-1}\mathbf{b}_{,z}^{(inv)} & = \tfrac{1}{8}\mathbf{T}_3^T\mathbf{b}_{,z}^{(inv)}\,. \end{array} \tag{33}$$

If these values are substituted to (21), we obtain

$$\begin{array}{l} \mathbf{r}_x = \tfrac{1}{4}\mathbf{T}_2^T \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \left(\tau_{xx}\mathbf{b}_{,x}^{(inv)} + \tau_{xy}\mathbf{b}_{,y}^{(inv)} + \tau_{xz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi \\ \mathbf{r}_y = \tfrac{1}{4}\mathbf{T}_2^T \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \left(\tau_{yx}\mathbf{b}_{,x}^{(inv)} + \tau_{yy}\mathbf{b}_{,y}^{(inv)} + \tau_{yz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi \\ \mathbf{r}_z = \tfrac{1}{4}\mathbf{T}_2^T \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \left(\tau_{zx}\mathbf{b}_{,x}^{(inv)} + \tau_{zy}\mathbf{b}_{,y}^{(inv)} + \tau_{zz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi \end{array} \tag{34}$$

where

$$\begin{aligned}
\mathbf{r}_x^{(inv)} &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\left(\tau_{xx}\mathbf{b}_{,x}^{(inv)}+\tau_{xy}\mathbf{b}_{,y}^{(inv)}+\tau_{xz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi\\
\mathbf{r}_y^{(inv)} &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\left(\tau_{yx}\mathbf{b}_{,x}^{(inv)}+\tau_{yy}\mathbf{b}_{,y}^{(inv)}+\tau_{yz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi\\
\mathbf{r}_z^{(inv)} &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\left(\tau_{zx}\mathbf{b}_{,x}^{(inv)}+\tau_{zy}\mathbf{b}_{,y}^{(inv)}+\tau_{zz}\mathbf{b}_{,z}^{(inv)}\right) Det\mathbf{J}\,\mathrm{d}\zeta\,\mathrm{d}\eta\,\mathrm{d}\xi
\end{aligned} \tag{35}$$

are the e-invariants of the restoring force vector. For the derivative of the vector elements $\mathbf{b}^{(inv)}$ with respect to x, y and z we can substitute

$$\mathbf{b}_{,x}^{(inv)} = \begin{bmatrix} 0\\ \xi_{,x}\\ \eta_{,x}\\ (\eta\xi)_{,x}\\ \zeta_{,x}\\ (\xi\zeta)_{,x}\\ (\eta\zeta)_{,x}\\ (\eta\xi\zeta)_{,x}\end{bmatrix} \quad \mathbf{b}_{,y}^{(inv)} = \begin{bmatrix} 0\\ \xi_{,y}\\ \eta_{,y}\\ (\eta\xi)_{,y}\\ \zeta_{,y}\\ (\xi\zeta)_{,y}\\ (\eta\zeta)_{,y}\\ (\eta\xi\zeta)_{,y}\end{bmatrix} \quad \mathbf{b}_{,z}^{(inv)} = \begin{bmatrix} 0\\ \xi_{,z}\\ \eta_{,z}\\ (\eta\xi)_{,z}\\ \zeta_{,z}\\ (\xi\zeta)_{,z}\\ (\eta\zeta)_{,x}\\ (\eta\xi\zeta)_{,z}\end{bmatrix} \tag{36}$$

while the Jacobian is now given as follows from (31)

$$\begin{aligned}
\mathbf{J} &= \begin{bmatrix} x_{,\xi} & x_{,\eta} & x_{,\zeta}\\ y_{,\xi} & y_{,\eta} & y_{,\zeta}\\ z_{,\xi} & z_{,\eta} & z_{,\zeta}\end{bmatrix}\\
&= \begin{bmatrix} x^{(0)} & x^{(1)} & x^{(2)} & x^{(12)} & x^{(3)} & x^{(13)} & x^{(23)} & x^{(123)}\\ y^{(0)} & y^{(1)} & y^{(2)} & y^{(12)} & y^{(3)} & y^{(13)} & y^{(23)} & y^{(123)}\\ z^{(0)} & z^{(1)} & z^{(2)} & z^{(12)} & z^{(3)} & z^{(13)} & z^{(23)} & z^{(123)}\end{bmatrix}\begin{bmatrix} 0 & 0 & 0\\ 1 & 0 & 0\\ 0 & 1 & 0\\ \eta & \xi & 0\\ 0 & 0 & 1\\ \zeta & 0 & \xi\\ 0 & \zeta & \eta\\ \eta\zeta & \xi\zeta & \eta\xi\end{bmatrix}.
\end{aligned} \tag{37}$$

Thus, there are two possibilities how to compute the local restoring force vector. They are represented by the relations (21) and (34).

Note that application of the relation (34) is very simple for the case of a unit cube element. As an example, let us consider a mesh of unit cubes into a part of the computational domain. Let this mesh be generated by the translation of the unit cube. Then the spatial transformation between the master element and arbitrary cube element becomes

$$\begin{aligned}
x &= \tfrac{1}{8}(x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8)+\xi &&= x^{(0)}+\xi\\
y &= \tfrac{1}{8}(y_1+y_2+y_3+y_4+y_5+y_6+y_7+y_8)+\eta &&= y^{(0)}+\eta\\
z &= \tfrac{1}{8}(z_1+z_2+z_3+z_4+z_5+z_6+z_7+z_8)+\zeta &&= z^{(0)}+\zeta.
\end{aligned} \tag{38}$$

It can be easily seen that for the Jacobian and its determinant in this case

$$\mathbf{J} = \begin{bmatrix} x_{,\xi} & x_{,\eta} & x_{,\zeta}\\ y_{,\xi} & y_{,\eta} & y_{,\zeta}\\ z_{,\xi} & z_{,\eta} & z_{,\zeta}\end{bmatrix} = \begin{bmatrix} 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} \qquad Det\mathbf{J} = 1 \tag{39}$$

holds and the derivative of the vector $\mathbf{b}^{(inv)}$ with respect to x, y and z has a very simple result

$$\begin{aligned} \mathbf{b}^{(inv)}_{,x} &= \left[0,1,0,\eta,0,\zeta,0,\eta\zeta\right]^T \\ \mathbf{b}^{(inv)}_{,y} &= \left[0,0,1,\xi,0,0,\zeta,\xi\zeta\right]^T \\ \mathbf{b}^{(inv)}_{,z} &= \left[0,0,0,0,1,\xi,\eta,\eta\xi\right]^T . \end{aligned} \tag{40}$$

Thus, using the formulations (32) and (33) we can compute the derivatives of displacements and the stress components for the square element as follows

$$\begin{aligned} u_{x,x} &= u_x^{(1)} + u_x^{(12)}\eta + u_x^{(13)}\zeta + u_x^{(123)}\eta\zeta \\ u_{x,y} &= u_x^{(2)} + u_x^{(12)}\xi + u_x^{(23)}\zeta + u_x^{(123)}\xi\zeta \\ u_{x,z} &= u_x^{(3)} + u_x^{(13)}\xi + u_x^{(23)}\eta + u_x^{(123)}\xi\eta \\ u_{y,x} &= u_y^{(1)} + u_y^{(12)}\eta + u_y^{(13)}\zeta + u_y^{(123)}\eta\zeta \\ u_{y,y} &= u_y^{(2)} + u_y^{(12)}\xi + u_y^{(23)}\zeta + u_y^{(123)}\xi\zeta \\ u_{y,z} &= u_y^{(3)} + u_y^{(13)}\xi + u_y^{(23)}\eta + u_y^{(123)}\xi\eta \\ u_{z,x} &= u_z^{(1)} + u_z^{(12)}\eta + u_z^{(13)}\zeta + u_z^{(123)}\eta\zeta \\ u_{z,y} &= u_z^{(2)} + u_z^{(12)}\xi + u_z^{(23)}\zeta + u_z^{(123)}\xi\zeta \\ u_{z,z} &= u_z^{(3)} + u_z^{(13)}\xi + u_z^{(23)}\eta + u_z^{(123)}\xi\eta. \end{aligned} \tag{41}$$

Now the local restoring force vector for the square element is obtained by exact evaluation of the integrals (34). The result is very simple

$$\mathbf{r}_x = \mathbf{T}_3^T \begin{bmatrix} \lambda(u_x^{(1)} + u_y^{(2)} + u_z^{(3)}) + 2\mu u_x^{(1)} \\ \mu(u_x^{(2)} + u_y^{(1)}) \\ \mu u_x^{(12)} + \lambda(u_x^{(12)} + u_z^{(23)})/3 \\ \mu(u_z^{(1)} + u_x^{(3)}) \\ \mu u_x^{(13)} + \lambda(u_x^{(13)} + u_y^{(23)})/3 \\ \mu(u_z^{(12)} + u_y^{(13)} + u_x^{(23)} + u_x^{(23)})/3 \\ (\lambda + 4\mu)u_y^{(123)}/9 \end{bmatrix} \tag{42}$$

$$\mathbf{r}_y = \mathbf{T}_3^T \begin{bmatrix} \mu(u_y^{(1)} + u_x^{(2)}) \\ \lambda(u_x^{(1)} + u_y^{(2)} + u_z^{(3)}) + 2\mu u_y^{(2)} \\ \mu u_y^{(12)} + \lambda(u_y^{(12)} + u_z^{(13)})/3 \\ \mu(u_y^{(3)} + u_z^{(2)}), \\ \mu(u_z^{(12)} + u_x^{(23)} + u_y^{(13)} + u_y^{(13)})/3 \\ \mu u_y^{(23)} + \lambda(u_x^{(13)} + u_y^{(23)})/3 \\ (\lambda + 4\mu)u_y^{(123)}/9 \end{bmatrix}$$

$$\mathbf{r}_z = \mathbf{T}_3^T \begin{bmatrix} \mu(u_x^{(3)} + u_z^{(1)}) \\ \mu(u_y^{(3)} + u_z^{(2)}) \\ \mu(u_y^{(13)} + u_x^{(23)} + u_z^{(12)} + u_z^{(12)})/3 \\ \lambda(u_x^{(1)} + u_y^{(2)} + u_z^{(3)}) + 2\mu u_z^{(3)} \\ \mu u_z^{(13)} + \lambda(u_y^{(12)} + u_z^{(13)})/3 \\ \mu u_z^{(23)} + \lambda(u_x^{(12)} + u_z^{(23)})/3 \\ (\lambda + 4\mu)u_z^{(123)}/9 \end{bmatrix}.$$

An algorithm efficiency is often measured by the number of arithmetical operations used in an algorithm. The number of required operations for the computation of the local restoring force vector for a hexahedral element H8 is given in the Table 1. During the

Case of algorithm	dividing	multiplying	add/substr.
Standard	8	2016	2092
Proposed (H8 element)	8	1728	1655
Proposed (cube element)	0	27	161

Tab. 1. Comparison of the number of arithmetical operations in 3D algorithms.

computation of the restoring force vectors $\mathbf{r}_x, \mathbf{r}_y$ the transformation of these vectors by the matrices $\mathbf{T}_2$ or $\mathbf{T}_3$ are needed. We remember, that the matrices $\mathbf{T}_n$, $n = 1, 2, 3$ contain only ± 1 values and therefore no multiplications are required in the matrix – vector multiplication. Moreover, the number of additions or subtractions required can be reduced because $\mathbf{T}_n$ can be written as a product of n very sparse matrices, that is, $\mathbf{T}_n = (\mathbf{H}_n)^n$, $n = 2, 3$ where

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad \mathbf{H}_3 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}. \tag{43}$$

The role of the e-invariants for is now briefly described in the next section.

4. THE E-INVARIANTS OF THE HEXAHEDRAL GEOMETRY

As mentioned already, the e-invariants save only the essential information needed for the computation of the local restoring force vector. Our approach is based on two groups of invariants – the e-invariants of the hexahedral geometry and the e-invariants of the displacements. Both forms of the restoring force vector computation (21) and

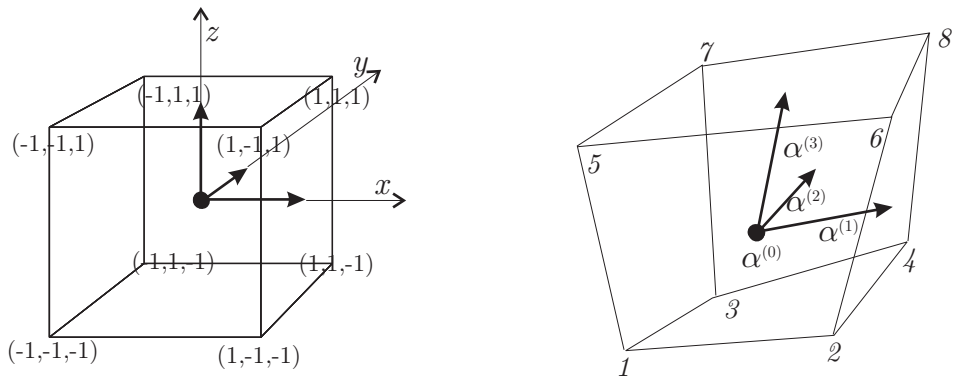

Fig. 2. Trilinear transformation.

(34) require to compute the derivatives of the spatial variables x, y, z and displacements u_x, u_y, u_z with respect to the ξ, η and ζ. It is easy to see from (31) that

$$
\begin{aligned}
x_{,\xi} &= 0x^{(0)} + 1x^{(1)} + 0x^{(2)} + \eta x^{(12)} + 0x^{(3)} + \zeta x^{(13)} + 0x^{(23)} + \eta\zeta x^{(123)} \\
x_{,\eta} &= 0x^{(0)} + 0x^{(1)} + 1x^{(2)} + \xi x^{(12)} + 0x^{(3)} + 0x^{(13)} + \zeta x^{(23)} + \xi\zeta x^{(123)} \\
x_{,\zeta} &= 0x^{(0)} + 0x^{(1)} + 0x^{(2)} + 0x^{(12)} + 1x^{(3)} + \xi x^{(13)} + \eta x^{(23)} + \eta\xi x^{(123)} \\
\\
y_{,\xi} &= 0y^{(0)} + 1y^{(1)} + 0y^{(2)} + \eta y^{(12)} + 0y^{(3)} + \zeta y^{(13)} + 0y^{(23)} + \eta\zeta y^{(123)} \\
y_{,\eta} &= 0y^{(0)} + 0y^{(1)} + 1y^{(2)} + \xi y^{(12)} + 0y^{(3)} + 0y^{(13)} + \zeta y^{(23)} + \xi\zeta y^{(123)} \\
y_{,\zeta} &= 0y^{(0)} + 0y^{(1)} + 0y^{(2)} + 0y^{(12)} + 1y^{(3)} + \xi y^{(13)} + \eta y^{(23)} + \eta\xi y^{(123)} \\
\\
z_{,\xi} &= 0z^{(0)} + 1z^{(1)} + 0z^{(2)} + \eta z^{(12)} + 0z^{(3)} + \zeta z^{(13)} + 0z^{(23)} + \eta\zeta z^{(123)} \\
z_{,\eta} &= 0z^{(0)} + 0z^{(1)} + 1z^{(2)} + \xi z^{(12)} + 0z^{(3)} + 0z^{(13)} + \zeta z^{(23)} + \xi\zeta z^{(123)} \\
z_{,\zeta} &= 0z^{(0)} + 0z^{(1)} + 0z^{(2)} + 0z^{(12)} + 1z^{(3)} + \xi z^{(13)} + \eta z^{(23)} + \eta\xi z^{(123)}
\end{aligned}
\tag{44}
$$

holds. The first e-invariant components $x^{(0)}, y^{(0)}$ and $z^{(0)}$ generate the point $\alpha^{(0)} = (x^{(0)}, y^{(0)}, z^{(0)})$ in the hexahedral domain. This point corresponds with the $(0, 0, 0)$ point into the master element in trilinear transformation shown in Figure 2. Coordinates of point $\alpha^{(0)}$ have the value

$$
\begin{aligned}
x^{(0)} &= \tfrac{1}{8}(x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8) \\
y^{(0)} &= \tfrac{1}{8}(y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8) \\
z^{(0)} &= \tfrac{1}{8}(z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8).
\end{aligned}
\tag{45}
$$

It is clear that $\alpha^{(0)}$ determines the location of the quadrilateral element in computational domain independently on the rotation and change of the element shape. Location $\alpha^{(0)}$ of the element is not needed for the computation of the restoring force vector. Therefore it also concerns the values $x^{(0)}, y^{(0)}, z^{(0)}$. The zero coefficients at (44) confirm this fact.

Another e-invariants are the vectors

$$\begin{aligned} \boldsymbol{\alpha}^{(1)} &= \left(x^{(1)}, y^{(1)}, z^{(1)}\right) \\ \boldsymbol{\alpha}^{(2)} &= \left(x^{(2)}, y^{(2)}, z^{(2)}\right) \\ \boldsymbol{\alpha}^{(3)} &= \left(x^{(3)}, y^{(3)}, z^{(3)}\right). \end{aligned} \tag{46}$$

These vectors correspond with the vectors $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ into the master element. Their coordinates are as follows

$$\begin{aligned} x^{(1)} &= \tfrac{1}{8}\left(-x_1 + x_2 - x_3 + x_4 - x_5 + x_6 - x_7 + x_8\right) \\ x^{(2)} &= \tfrac{1}{8}\left(-x_1 - x_2 + x_3 + x_4 - x_5 - x_6 + x_7 + x_8\right) \\ x^{(3)} &= \tfrac{1}{8}\left(-x_1 - x_2 - x_3 - x_4 + x_5 + x_6 + x_7 + x_8\right). \end{aligned} \tag{47}$$

It is easy to verify that these coordinates fulfill

$$\begin{aligned} x^{(1)} &= \tfrac{1}{4}\left(x_2 + x_4 + x_6 + x_8\right) - x^{(0)} \\ x^{(2)} &= \tfrac{1}{4}\left(x_3 + x_4 + x_7 + x_8\right) - x^{(0)} \\ x^{(3)} &= \tfrac{1}{4}\left(x_5 + x_6 + x_7 + x_8\right) - x^{(0)} \end{aligned} \tag{48}$$

and similarly for $y^{(1)}, y^{(2)}, y^{(3)}, z^{(1)}, z^{(2)}, z^{(3)}$. The positions of the vectors $\boldsymbol{\alpha}^{(1)}$, $\boldsymbol{\alpha}^{(2)}$ and $\boldsymbol{\alpha}^{(3)}$ are shown in the Figure 2. This vectors are independent on the location and rotation of the hexahedral element. The zero coefficients at the components $x^{(1)}, y^{(1)}, z^{(1)}$, $x^{(2)}, y^{(2)}, z^{(2)}$ and $x^{(3)}, y^{(3)}, z^{(3)}$ in (44) indicate the positions where they can be omitted during the computation.

The next invariants are vectors $\boldsymbol{\alpha}^{(12)} = (x^{(12)}, y^{(12)}, z^{(12)})$, $\boldsymbol{\alpha}^{(13)} = (x^{(13)}, y^{(13)}, z^{(13)})$, $\boldsymbol{\alpha}^{(23)} = (x^{(23)}, y^{(23)}, z^{(23)})$ and $\boldsymbol{\alpha}^{(123)} = (x^{(123)}, y^{(123)}, z^{(12)})$. which is always non-zero for a general case of hexahedral element. This vectors corresponds with the zero vector in the master element. The coordinates of this vector are

$$\begin{aligned} x^{(12)} &= \tfrac{1}{8}\left(+x_1 - x_2 - x_3 + x_4 + x_5 - x_6 - x_7 + x_8\right) \\ x^{(13)} &= \tfrac{1}{8}\left(+x_1 - x_2 + x_3 - x_4 - x_5 + x_6 - x_7 + x_8\right) \\ x^{(23)} &= \tfrac{1}{8}\left(+x_1 + x_2 - x_3 - x_4 - x_5 - x_6 + x_7 + x_8\right) \\ x^{(123)} &= \tfrac{1}{8}\left(-x_1 + x_2 + x_3 - x_4 + x_5 - x_6 - x_7 + x_8\right) \end{aligned} \tag{49}$$

or

$$\begin{aligned} x^{(12)} &= \tfrac{1}{4}\left(x_1 + x_4 + x_5 + x_8\right) - x^{(0)} \\ x^{(13)} &= \tfrac{1}{4}\left(x_1 + x_3 + x_6 + x_8\right) - x^{(0)} \\ x^{(23)} &= \tfrac{1}{4}\left(x_1 + x_2 + x_7 + x_8\right) - x^{(0)} \\ x^{(123)} &= \tfrac{1}{4}\left(x_2 + x_3 + x_5 + x_8\right) - x^{(0)} \end{aligned} \tag{50}$$

and similarly for yth and zth invariants. This value is independent from the element location and rotation and can be used as measures of deformation of the element. The derivatives of the displacements u_x, u_y and u_z with respect to ξ, η and ζ can be derived according to equation (32). The values

$$u_x^{(0)} = \frac{u_{x1} + u_{x2} + u_{x3} + u_{x4} + u_{x5} + u_{x6} + u_{x7} + u_{x8}}{8} \tag{51}$$

and similarly $u_y^{(0)}, u_z^{(0)}$ represent mean value of displacements. In other words they express the global shifting of the element, but the restoring force vector is not dependent on these values.

Interesting is the physical interpretation of values $u_x^{(1)}, u_x^{(2)}, u_x^{(3)}$ and similarly displacements in the yth and zth direction on cube element. The values

$$u_x^{(1)} = \frac{\frac{u_{x2}+u_{x4}+u_{x6}+u_{x8}}{4} - \frac{u_{x1}+u_{x3}+u_{x5}+u_{x7}}{4}}{2}$$
$$u_x^{(2)} = \frac{\frac{u_{x3}+u_{x4}+u_{x7}+u_{x8}}{4} - \frac{u_{x1}+u_{x2}+u_{x5}+u_{x6}}{4}}{2} \tag{52}$$
$$u_x^{(3)} = \frac{\frac{u_{x5}+u_{x6}+u_{x7}+u_{x8}}{4} - \frac{u_{x1}+u_{x2}+u_{x3}+u_{x4}}{4}}{2}$$

represent the finite difference approximation of displacements in x,y and z direction for the element with length of site 2. Similarly, $u_y^{(1)}$, $u_y^{(2)}, u_y^{(3)}$ and $u_z^{(1)}$, $u_z^{(2)}, u_z^{(3)}$ represent the finite difference approximation of displacements in y and z direction.

The values

$$r_x^{(0)} = \frac{r_{x1} + r_{x2} + r_{x3} + r_{x4} + r_{x5} + r_{x6} + r_{x7} + r_{x8}}{8}, \tag{53}$$

$r_y^{(0)}$ and $r_z^{(0)}$ represent the global force acting on the element, but the local forces are not dependent on these values and therefore they are not needed for computation.

5. CONCLUSION

We developed a new method of the restoring force vector computation in 2D and 3D case of the FEM computation for a elastodynamics equation. The new method reduces the number of arithmetical operation by $\frac{1}{3}$ approximately, while keeping the same memory requirements. The main idea of the speed-up is the use of new base functions that allow the use of the e-invariants. These e-invariants have such property that they save only the essential information needed for the computation of the local restoring force vector and redundant information is omitted from the computation.

ACKNOWLEDGEMENT

This work was supported by grant APVV-0184-10.

(Received March 2, 2012)

REFERENCES

[1] R. J. Archuleta: Experimental and Numerical Three-Dimensional Simulations of Strike-Slip Earthquakes. PhD. Thesis. University of California, San Diego 1976.

[2] M. Balazovjech and L. Halada: Effective computation of restoring force vector in FEM. Kybernetika *43* (2007), 6, 767–776.

[3] M. Balazovjech: Efektivny vypocet seizmickeho pohybu metodou konecnych elementov. Dissertation, FMPH Bratislava 2008.

[4] T. Belytscho, W. K. Liu, and B. Moran: Nonlinear Finite Elements for Continua and Structures. John Wiley and Sons, 2000.

[5] C. A. Felippa: Introduction to Finite Element Methods. Lecture Notes. University of Colorado, Boulder 2005.

[6] G. A. Frazier and C. M. Petersen: 3-D stress wave code for the Illiac IV. Systems. Systems, Science and Software Report SSS-R-74-2103, 1974.

[7] T. J. R. Hughes: The Finite Element Method. Linear Static and Dynamic Finite Element Method Analysis. Prentice Hall, 2000.

[8] P. Moczo, J. Kristek, M. Galis, P. Pazak, and M. Balazovjech: The finite-difference and finite-element modeling of seismic wave propagation and earthquake motion. Acta Phys. Slovaca *57* (2007), 2, 177–406.

[9] J. N. Reddy: An Introduction to the Finite Element Method. McGraw-Hill, New York 1993.

[10] G. Strang and G. J. Fix: An Analysis of the Finite Element Method. Wellesley Cambridge Press 1988.

[11] R. Taborda, J. Lopez, H. Karaoglu, J. Urbanic, and J. Bielak: Speeding up Finite Element Wave Propagation for Large-scale Earthquake Simulations. Technical Report CMU-PDL-10-109, Carnegie Mellon University, Parallel Data Lab. 2010.

Martin Balazovjech, Department of Mathematics, Slovak University of Technology, Radlinského 11, 813 68 Bratislava. Slovak Republic.
e-mail: balazovjech@math.sk

Ladislav Halada, Institute of Informatics, Slovak Academy of Sciences, Dúbravská cesta 9, 842 28 Bratislava. Slovak Republic.
e-mail: ladislav.halada@savba.sk