Computational aspects of generalized continua based on moving least square approximations

Sebastian Skatulla

Thesis submitted to the University of Adelaide
for the degree of Doctor of Philosophy

September, 2006

School of Mechanical Engineering
The University of Adelaide

Appendix A

Customizing a spline with a specific continuity

A great benefit of splines in general is that they can be designed to fit a given set of sample points, but also meet specific continuity requirements. The latter is achieved by incorporating higher order polynomials which provide the desired continuity. However, higher order polynomials exhibit a tendency to oscillate, a behaviour which is not desired for the weight function. An essential specification on a weight function is namely, besides the higher order continuity, its compact support. That is, it has to have a clear maximum in the center from which it rapidly and monotonically converges to zero. The latter means without oscillations. In order to prevent the oscillations caused by the use of the involved higher order polynomials, the definition space of the weight function, which is its compact support ω, is separated into several subspaces $\omega^{(s)}$. Within each subspace the spline is defined by a different set of polynomial coefficients $\mathbf{a}^s$. These subspaces are linked with each other via boundary conditions as in the continuity of the spline and its higher order derivatives throughout ω.

In this thesis a C^3-continuous spline based on a fourth order Pascal-type polynomial $\mathbf{P}$ (Eq. 3.37) is developed by a scheme as outlined in the following. Assume that the definition space of the spline or support ω is parameterized by the coordinate r which takes values out of the interval $[-1,\ 1]$. The spline's definition space is subsequently split into six subspaces $\omega^{(s)} := [g_s,\ g_{s+1}]$, $s = 1, 6$. The limits of these subspaces g_i are given by the following seven sample points: $g_1 = (-1.0;\ 0)$, $g_2 = (-2.0/3.0;\ 0.05)$, $g_3 = (-1.0/3.0;\ 0.5)$, $g_4 = (0;\ 1.0)$, $g_5 = (1.0/3.0; 0.5)$, $g_6 = (2.0/3.0;\ 0.05)$ and $g_7 = (1.0; 0)$. The spline has to interpolate these sample points exactly and they are therefore at the same time boundary conditions on the subspace $\omega^{(s)}$. Furthermore, we consider the following boundary conditions on the subspaces which ensure the spline's continuity which is wanted to be C^3:

- C^0 continuity between $\omega^{(1)}$ and $\omega^{(2)}$: $\mathbf{P}(g_2(1)) \cdot \mathbf{a}^{(1)} - \mathbf{P}(g_2(1)) \cdot \mathbf{a}^{(2)} = 0$
- C^1 continuity between $\omega^{(1)}$ and $\omega^{(2)}$: $\mathbf{P}_{,r}(g_2(1)) \cdot \mathbf{a}^{(1)} - \mathbf{P}_{,r}(g_2(1)) \cdot \mathbf{a}^{(2)} = 0$
- C^2 continuity between $\omega^{(1)}$ and $\omega^{(2)}$: $\mathbf{P}_{,rr}(g_2(1)) \cdot \mathbf{a}^{(1)} - \mathbf{P}_{,rr}(g_2(1)) \cdot \mathbf{a}^{(2)} = 0$

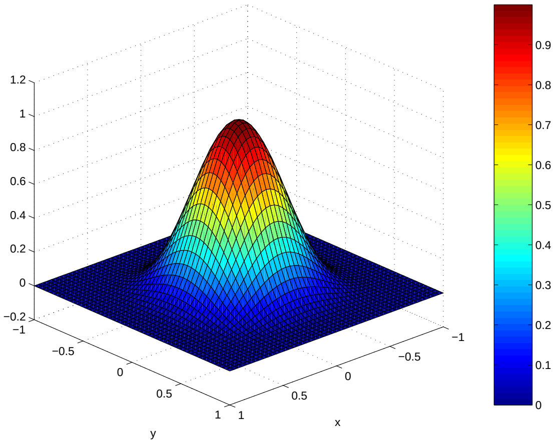

Figure A.1: *C^3 quartic spline $\Phi = w\,w$*

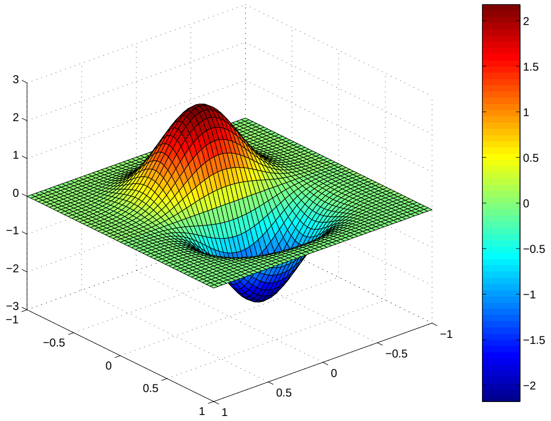

Figure A.2: *first order derivative of the C^3 quartic spline $\Phi_{,y} = \frac{1}{\varrho}\,w\,w_{,y}$*

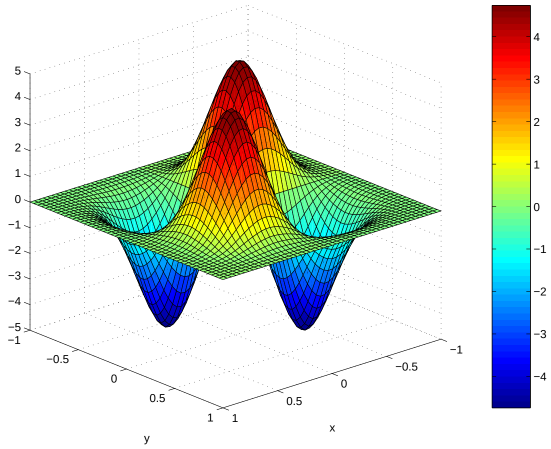

Figure A.3: *second order derivative of the C^3 quartic spline $\Phi_{,xy} = \frac{1}{\varrho^2}\,w_{,x}\,w_{,y}$*

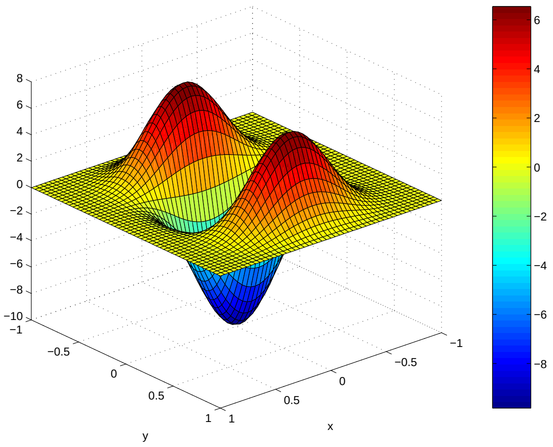

Figure A.4: *second order derivative of the C^3 quartic spline $\Phi_{,yy} = \frac{1}{\varrho^2}\,w\,w_{,yy}$*

- C^3 continuity between $\omega^{(1)}$ and $\omega^{(2)}$: $\mathbf{P}_{,rrr}(g_2(1))\cdot\mathbf{a}^{(1)} - \mathbf{P}_{,rrr}(g_2(1))\cdot\mathbf{a}^{(2)} = 0$
- C^0 continuity between $\omega^{(2)}$ and $\omega^{(3)}$: $\mathbf{P}(g_3(1))\cdot\mathbf{a}^{(2)} - \mathbf{P}(g_3(1))\cdot\mathbf{a}^{(3)} = 0$
- C^1 continuity between $\omega^{(2)}$ and $\omega^{(3)}$: $\mathbf{P}_{,r}(g_3(1))\cdot\mathbf{a}^{(2)} - \mathbf{P}_{,r}(g_3(1))\cdot\mathbf{a}^{(3)} = 0$
- C^2 continuity between $\omega^{(2)}$ and $\omega^{(3)}$: $\mathbf{P}_{,rr}(g_3(1))\cdot\mathbf{a}^{(2)} - \mathbf{P}_{,rr}(g_3(1))\cdot\mathbf{a}^{(3)} = 0$
- C^3 continuity between $\omega^{(2)}$ and $\omega^{(3)}$: $\mathbf{P}_{,rrr}(g_3(1))\cdot\mathbf{a}^{(2)} - \mathbf{P}_{,rrr}(g_3(1))\cdot\mathbf{a}^{(3)} = 0$
- C^0 continuity between $\omega^{(3)}$ and $\omega^{(4)}$: $\mathbf{P}(g_4(1))\cdot\mathbf{a}^{(3)} - \mathbf{P}(g_4(1))\cdot\mathbf{a}^{(4)} = 0$
- C^1 continuity between $\omega^{(3)}$ and $\omega^{(4)}$: $\mathbf{P}_{,r}(g_4(1))\cdot\mathbf{a}^{(3)} - \mathbf{P}_{,r}(g_4(1))\cdot\mathbf{a}^{(4)} = 0$
- C^2 continuity between $\omega^{(3)}$ and $\omega^{(4)}$: $\mathbf{P}_{,rr}(g_4(1))\cdot\mathbf{a}^{(3)} - \mathbf{P}_{,rr}(g_4(1))\cdot\mathbf{a}^{(4)} = 0$

- C^3 continuity between $\omega^{(3)}$ and $\omega^{(4)}$: $\mathbf{P}_{,rrr}(g_4(1)) \cdot \mathbf{a}^{(3)} - \mathbf{P}_{,rrr}(g_4(1)) \cdot \mathbf{a}^{(4)} = 0$
- C^0 continuity between $\omega^{(4)}$ and $\omega^{(5)}$: $\mathbf{P}(g_5(1)) \cdot \mathbf{a}^{(4)} - \mathbf{P}(g_5(1)) \cdot \mathbf{a}^{(5)} = 0$
- C^1 continuity between $\omega^{(4)}$ and $\omega^{(5)}$: $\mathbf{P}_{,r}(g_5(1)) \cdot \mathbf{a}^{(4)} - \mathbf{P}_{,r}(g_5(1)) \cdot \mathbf{a}^{(5)} = 0$
- C^2 continuity between $\omega^{(4)}$ and $\omega^{(5)}$: $\mathbf{P}_{,rr}(g_5(1)) \cdot \mathbf{a}^{(4)} - \mathbf{P}_{,rr}(g_5(1)) \cdot \mathbf{a}^{(5)} = 0$
- C^3 continuity between $\omega^{(4)}$ and $\omega^{(5)}$: $\mathbf{P}_{,rrr}(g_5(1)) \cdot \mathbf{a}^{(4)} - \mathbf{P}_{,rrr}(g_5(1)) \cdot \mathbf{a}^{(5)} = 0$
- C^0 continuity between $\omega^{(5)}$ and $\omega^{(6)}$: $\mathbf{P}(g_6(1)) \cdot \mathbf{a}^{(5)} - \mathbf{P}(g_6(1)) \cdot \mathbf{a}^{(6)} = 0$
- C^1 continuity between $\omega^{(5)}$ and $\omega^{(6)}$: $\mathbf{P}_{,r}(g_6(1)) \cdot \mathbf{a}^{(5)} - \mathbf{P}_{,r}(g_6(1)) \cdot \mathbf{a}^{(6)} = 0$
- C^2 continuity between $\omega^{(5)}$ and $\omega^{(6)}$: $\mathbf{P}_{,rr}(g_6(1)) \cdot \mathbf{a}^{(5)} - \mathbf{P}_{,rr}(g_6(1)) \cdot \mathbf{a}^{(6)} = 0$
- C^3 continuity between $\omega^{(5)}$ and $\omega^{(6)}$: $\mathbf{P}_{,rrr}(g_6(1)) \cdot \mathbf{a}^{(5)} - \mathbf{P}_{,rrr}(g_6(1)) \cdot \mathbf{a}^{(6)} = 0$

Furthermore, the spline has to converge to zero on its left and right outskirts:

- boundary condition on $\omega^{(1)}$: $\mathbf{P}(g_1(1)) \cdot \mathbf{a}^{(1)} = g_1(2)$
- boundary condition on $\omega^{(6)}$: $\mathbf{P}(g_7(1)) \cdot \mathbf{a}^{(6)} = g_7(2)$

and has to have an absolute minimum there:

- boundary condition on $\omega^{(1)}$: $\mathbf{P}_{,r}(g_1(1)) \cdot \mathbf{a}^{(1)} = 0$
- boundary condition on $\omega^{(1)}$: $\mathbf{P}_{,rr}(g_1(1)) \cdot \mathbf{a}^{(1)} = 0$
- boundary condition on $\omega^{(1)}$: $\mathbf{P}_{,rrr}(g_1(1)) \cdot \mathbf{a}^{(1)} = 0$
- boundary condition on $\omega^{(6)}$: $\mathbf{P}_{,r}(g_7(1)) \cdot \mathbf{a}^{(6)} = 0$
- boundary condition on $\omega^{(6)}$: $\mathbf{P}_{,rr}(g_7(1)) \cdot \mathbf{a}^{(6)} = 0$
- boundary condition on $\omega^{(6)}$: $\mathbf{P}_{,rrr}(g_7(1)) \cdot \mathbf{a}^{(6)} = 0$

Finally, the spline has to exactly interpolate a certain given value in the domain center and has to have a maximum there:

- boundary condition on $\omega^{(3)}$: $\mathbf{P}(g_4(1)) \cdot \mathbf{a}^{(3)} = g_4(2)$
- boundary condition on $\omega^{(3)}$: $\mathbf{P}_{,r}(g_4(1)) \cdot \mathbf{a}^{(3)} = 0$

These boundary conditions provide 30 equations which are assembled in an algebraic equation system:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}\,, \tag{A.1}$$

where vector $\mathbf{x}$ contains the unknown polynomial coefficients $\mathbf{a}^{(s)}$. After solving this equation system a C^3-continuous quartic spline is obtained which consists of six definition spaces $\omega^{(s)}$. For the two dimensional space we define $r_x = \frac{x-x_0}{\varrho}$ and $r_y = \frac{y-y_0}{\varrho}$, where $(x_0;\, y_0)$ denotes the center point of the spline. Then the spline, its first and second order derivatives are illustrated for two dimensions in Fig. A.1, Fig. A.2, Fig. A.3 and Fig. A.4, respectively.

Appendix B

Parallelization

Generally, the use of a MLS-based meshfree code is much more costly than FEM-based one. That is, the MLS shape functions and their derivatives have to be computed individually for every integration point and every particle. Furthermore, each integration point is supported by a larger number of particles than in FEM which increases the computing time during numerical integration over the problem domain, but also when solving the resulting discrete equation system. The latter is due to the larger bandwidth of the coefficient matrix of the discrete equation system. The variational formulations presented in this thesis are partly very involved so that the evaluation consumes a large amount of computation time in general, and even more for meshfree methods due to the more extensive particle support. Therefore, it is advisable to parallelize the implementation of the meshfree code so that the modelling can be undertaken on a supercomputer with a distributed memory architecture.

A parallel supercomputer consists of a certain number of CPUs so-called *nodes*, where each node has its own independent memory. The computation load and the needed memory is more or less equally distributed over the used nodes. In that way a simulation task is split into a pre-defined number of processes or so-called *threads*. Each node hosts one or more threads depending on whether the node is a single- or a multi-processor CPU. Usually, there is some interaction between the nodes resulting in exchange of data from node to node over a high-speed network which connects the nodes. Between the threads on a single multi-processor node no network communication is necessary, because they share their memory with each other. The transfer of data between nodes is initiated by special commands in the code that make use of the so-called *message parsing interface* (MPI) which must be linked into the code as external library. The amount of communication between the nodes however, has to be minimized, because it can otherwise significantly slow down the calculation. Therefore, when implementing the code, it must be carefully considered which algorithms are large enough so that the performance gain dealing with them in parallel, is not taken up again by firstly distributing those tasks over the network to all nodes and secondly merging the results after completion. The enforcement of the essential boundary conditions outlined in Sec. 3.3.2 for an example, is basically a simple matrix multiplication which need not be computed in parallel. However, as the coefficient matrix of the discrete equation is distributed over the nodes this task has to be done in parallel. Furthermore, it is essential that all nodes have assigned an equal share of the computational load, because during the modelling the processes have to be synchronized several times. This means that all processes are stopped, until each of them has reached a certain point of the calculation process. An unequal load results in some nodes running idle, while waiting for the others. An example for this is the assembling of the discrete equation system which must have been completed by all threads, before it can be solved. The maximum number of nodes is limited by the fact that with an increasing number of nodes the total amount of communication between the nodes is rising as well. Therefore, for a problem of a certain size, there is a maximum number of nodes which provide a performance gain compared to a single workstation.

In this work a parallelized MLS-based meshfree modelling software is implemented using the *C++* programming language. *C++* is an object-oriented language which makes it easy to structure and extend the code, e.g. each integration point and each particle represent a separate object which can carry various properties such as coordinates, integration weights, its identity number and history variables. Furthermore, it features a flexible, dynamical memory allocation and various so-called standard libraries that facilitate the coding and are already optimized with respect to numerical performance. Additionally, diverse software libraries are available for *C++* such as equation solvers and graph partitioner, the use of which is described in the later.

For quasi-static problems the basic structure of the code is now outlined in the following:

1) Reading in of all needed problem parameters and a FEM background mesh

At first all needed problem parameters and a FEM background mesh defining the problem domain are read in. As already mentioned, a FEM mesh provides the particle distribution and the background grid, where the numerical integration - the *Gauss quadrature* - takes place. All particles are stored at all processes, because all threads need to access all particles, e.g. determining the particle influence zones which is described in the latter. For large scale problems however, it would be necessary to store the particles distributed over the nodes, otherwise the memory limits of a single node would be exceeded. The element-particle connectivity list is also temporarily stored as a whole at all processes.

2) Sorting the particles and assigning their identify numbers

The coefficient matrix of the discrete equation system is desired to be band-structured so that the solving of the equation system can be performed most efficiently. Since the line and column numbers of the coefficient matrix correspond to the particle parameters or its degrees of freedom, respectively, it must be ensured that the identity number of the particles are assigned subsequently and corresponding to the particles' closest neighbouring particles. This is as a particle interacts only with those particles in its vicinity. Reciprocally, only these particle parameters are connected and result in entries in the coefficient matrix. Therefore, the particles are sorted by their coordinates with respect to all three coordinated directions. Then the identity numbers of all particles and their parameters can be assigned correspondingly to this sorting order. Assuming particle P_I has got n parameters d_J^I then the parameter identity numbers are determined as follows

$$d_J^I = I \cdot n + J \,. \tag{B.1}$$

3) Distribution of the FEM elements over the allocated threads

The FEM elements which are the integration cells are evenly distributed over all allocated threads so that accordingly, each process computes later for an equal portion of quadrature cells 1) the integration points and weights, 2) the particle weights and 3) its share on the evaluation of the continuous integral over the problem. The distribution is achieved by sorting the elements correspondingly to their position with respect to one coordinate direction and splitting them in equal portions for each process.

4) Computation of the Gauss quadrature coordinates and weights as well the particle weights

Using FEM ansatz-functions for tetrahedral or hexahedral elements and triangle or quadrilateral elements, respectively, each process determines for its share of integration cells the coordinates and the weights of the integration points as well as as the particle weights according to (Eq. 3.20). Afterwards, the FEM element-particle connectivity list is not needed anymore and can be deleted.

5) Determination of the particle influence zones

An essential precondition to calculate the MLS approximation functions (Eq. 3.5) is that the moment matrix (Eq. 3.4) can be inverted. Accordingly, if a basis polynomial of m-th order is used to approximate a function at a point x, then the influence zones of the closest particles must be determined in such a way that the weighted least square fit (Eq. 3.2) is evaluated on at least m sample points which are the particles. That is, the weight-functions associated with these neighbouring particles have to cover point x considering that these weight-functions define each particles domain of influence.

Generally, point x is either an integration point or a particle, where the solution needs to be approximated. In this work, first and second order basis polynomials of the Pascal type are applied. In order to provide a stable solution approximation throughout the domain it is found that each particle has to support at least two neighbouring particles in each positive and negative coordinate direction. An irregular particle distribution can require three or four neighbouring particles.

Since the influence zones are cuboids, the influence radii for all three coordinate directions are separately determined by a special searching algorithm. Each process deals with an equal portion of particles. After completion the results are exchanged among all threads. Therefore, each process possesses a complete set of particles with their associated influence radii.

6) Distribution of all Gauss quadrature points over the allocated threads

Using a distributed memory architecture supercomputer, larger arrays are usually stored in distributed manner over the nodes. This is especially the case for the coefficient matrix of the discrete equation system. In order to avoid large scale network communication between the threads, it is necessary to store the quadrature points accordingly they generate entries for certain parts of the coefficient matrix.

For a simple problem domain geometry it is sufficient that the volume and boundary integration points stay at those processes, where their coordinates and weights have been determined earlier on. This basically means that the sorting and distribution of the quadrature cells is adopted.

In case of more complex geometrical configuration such as bodies containing voids or cracks it is meaningful to make use of a special partitioning algorithm. By respecting the connectivity of the integration points to each other, it can be ensured that the amount of communication is minimized and the computational load is equally distributed.

In this work the external partitioning library *ParMETIS* (Karypis and Kumar 1997; Karypis and Kumar 1998) is incorporated which features a so-called *graph partitioner*. The implemented algorithms in this library are based on the *multilevel recursive-bisection*, *multilevel k-way*, and *multi-constraint* partitioning schemes. This kind of partitioning procedure utilizes a so-called graph which represents a list of so-called *vertices* connected with so-called *edges*. All vertices and edges have associated weights, where the vertex weights address the amount of connectivity to neighbouring vertices and the edge weights the distance to each other. Generally, either the vertex weights or the edge weights can be active.

In the meshfree code each integration point stands for a vertex and its neighbouring integration points are the edges. The graph is set up by finding for each integration point those particles which support it. As edges we have all the other integration points which are also supported by the same particles. Using the vertex weights, the partitioning algorithm is optimized with respect to the computational load each thread will have assigned. Using the edge weights, the amount of communication will be minimized which arises during the evaluation of the integration points. Since the computing time consumed by the numerical integration over the elaborate formulations developed in this thesis exceeds the amount of communication time by far, the use of vertex weights is advisable. For bigger problems the graphs quickly reaches an immense size which directly affects the overall execution time of the partition algorithm and the memory needed to store it. It is therefore imperative to reduce the connectivity lists of each integration points to its closest neighbours.

As a result, the graph partitioner provides for each process its share of the integration points as well as the so-called *edgecut* which gives information of the amount of communication arising by this partition scheme. The edgecut means in this context that integration points lying close to a partition boundary but on different sides i.e. in different partitions, are supported by a certain number of the same particles. Consequently, the influence zones of those particles exhibit intersection and are therefore interacting across the partition boundaries depicted in Fig. B.1. Out of experience it is found that only 10 % of the original graph size is sufficient to achieve excellent results. That is, each vertex is only connected with four of its closest edges. This is plausible as the edges in the closest vicinity exhibit the greatest level of connectivity to the vertex.

Obviously, the generation of the initial graph is a very elaborate task and other partition algorithms only need the coordinates of the vertices to operate, but not a connectivity list. These kind of procedures however, exclusively account for the geometrical closeness. In case of voids within the problem domain, vertices lying at opposite sides are not recognized as not being linked with each other. Hence, these methods have only limited usage in

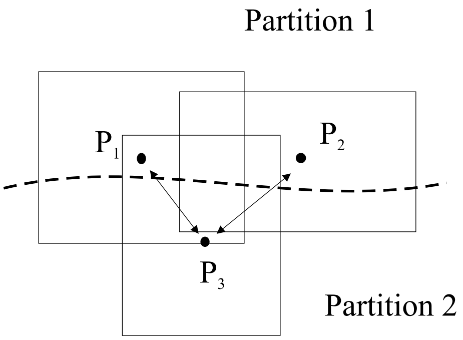

Figure B.1: *particles support across the partition boundary*

such cases.

When the partitioning is finished and the quadrature points are re-distributed according to the partition scheme, then for each integration point are those particles determined which support it. Each integration point is a *C++* object and stores its own, individual particle support list as an element of this *C++* object.

7) Computation of the shape functions and their derivatives at all integration points and particles

The MLS-approximation of the solution function and its derivatives requires that at each integration point and each particle the ordinates of the shape function and its derivatives of the supporting particles are computed. Note that the MLS approximation scheme at the particles only needs to be calculated, if the results of modelling is plotted at the particles. Parallelizing this task, each process obtains for its share of the integration points and particles the shape functions and their derivatives of the supporting particles according to (Eq. 3.5). Each integration point and particle is a separate *C++* object and stores the corresponding local MLS-approximation scheme as a *C++* object element.

8) Calculation of the modified shape functions and their derivatives

If the essential boundary conditions are enforced by the modified boundary collocation method described in Sec. 3.3.2 the approximation functions of those particles have to be modified which influence boundary particles having essential boundary conditions applied and also those particles which influence the earlier. The method involves the inversion of a matrix containing the unmodified shape functions ordinates (Eq. 3.66). Generally, this part of the code could be parallelized, but as mentioned in Sec. 3.3.2 this boundary enforcement procedure is as a whole not suitable for large scale problems anyway.

9) Determination of a local sparse storage scheme of the coefficient matrix

The coefficient matrix of the discrete equation system is generally only sparsely set. Consequently, it is meaningful to store only those entries which are different from zero and which are at a thread locally computed. The terminology *local* refers to the local part of a *global* entity which is assigned to a certain process only. Merging the local parts results in the corresponding global entity. Note however, the distribution of the global entity over the threads is achieved in such a way that we find a limited amount of redundancy and intersection. Now, it is necessary to obtain for each local quadrature point $\mathbf{x}_G$ a set of particles Λ_{x_G} which support this point

$$\Lambda_{x_G} = \left\{ I \,\middle|\, \mathbf{x}_G \in \omega_I \right\} , \tag{B.2}$$

where ω_I is the influence zone associated with the particle P_I. All particles $P_I \in \Lambda_{x_G}$ and correspondingly their particle parameters are in interaction with each other. Accordingly, the local coefficient matrix of the discrete equation system will have entries different from zero which relate to these particle parameters. This is clear as for all particles $P_I \in \Lambda_{x_G}$ and $P_J \in \Lambda_{x_G}$, the following holds

$$\omega_I \cap \omega_J \neq \emptyset \, . \tag{B.3}$$

In this way each node holds its local part or more precisely, a local portion of lines of the global coefficient matrix, where we have

$$\mathbf{K}^{global} = \sum_p^n \mathbf{K}^{lokal,p} , \qquad \text{with } n = \text{number of processes} \, . \tag{B.4}$$

Correspondingly, each node needs to allocate memory only for its local share of the coefficient matrix. The redundancy of matrix entries on different processes leads to communication between the processes, a fact is which will be addressed under Point 10. For larger problems the memory requirement of the global coefficient matrix would quickly exceed the memory capacity of a single node. The distribution of the global coefficient matrix over the nodes is therefore absolutely imperative.

Note that the use of the modified boundary collocation method outlined in Sec. 3.3.2 significantly increases the size of the sparse coefficient matrix. As reason can be found that using the modified boundary collocation method, the set Λ_{x_G} does not only contain the identity numbers of particles that directly influence each other, but also in case of particles having essential boundary conditions applied, all their neighbouring particles as well, even if the latter do not support $\mathbf{x}_G$.

In the code of this work a sparse matrix storage scheme is applied which is well-known in the literature (Bathe 1982). Hereby, only the band-structure of the thread's share on the coefficient matrix is stored as a vector. It is therefore necessary to store for each line of the local coefficient matrix the position of the first and last element within this vector.

The evaluation of the local share of integration points also results in a local internal and external force vector which are similar to the local coefficient matrix only partly set. It is therefore only as much memory for both allocated as necessary.

10) Computation of the coefficient matrix, the internal and the external force vector

The problem is now evaluated at each local integration point which involves the approximation of the kinematical quantities, e.g. the displacement vector, the deformation gradient and the strain tensor, and subsequently, the calculation of constitutive quantities such as the tangent and the stress. As a result we achieve at each integration point a part of the coefficient matrix $\mathbf{K}$, the internal and external force vector $\mathbf{f}_{int}$ and $\mathbf{f}_{ex}$, respectively, which correspond to the degrees of freedom of its supporting particles. When the whole local portion of the integration points have been computed, each process holds a local share of the discrete equation system:

$$\mathbf{K}^{local}\,\mathbf{d}^{local} = \mathbf{f}^{local}_{ext} - \mathbf{f}^{local}_{int}\,, \tag{B.5}$$

where vector $\mathbf{d}^{local}$ contains the local portion of the increments of particle parameters Δu_I. Now, it is important to realize that integration points near the partition boundaries partly provides values for the same global coefficient matrix entries K^{global}_{ij}, as well as for $f^{global}_{ext,i}$ and $f^{global}_{int,i}$. Therefore, after the numerical integration of the problem has been completed at each thread, those lines of the global discrete equation system which are partly computed at several nodes have to be merged according to the following expression

$$K^{global}_{ij} = \sum_{p}^{n} K^{lokal,p}_{ij}\,, \qquad \text{with } n = \text{number of processes}\,, \tag{B.6}$$

$$f^{global}_{ext,i} = \sum_{p}^{n} f^{local,p}_{ext,i}\,, \qquad \text{with } n = \text{number of processes}\,, \tag{B.7}$$

$$f^{global}_{int,i} = \sum_{p}^{n} f^{local,p}_{int,i}\,, \qquad \text{with } n = \text{number of processes}\,, \tag{B.8}$$

and exclusively assigned to one single process - a so-called *root* process. This task requires communication between the nodes i.e. data exchange. Hence, before solving the discrete equation each thread owns a certain part of the discrete equation system or more precisely, a certain portion of lines without redundancy.

Hereby, it is noted that the amount of communication is usually less for FEM, because the particle interactivity is less also. This holds especially for an irregular meshfree particle distribution, where the particle connectivity can be partly more than twice as much as in FEM.

11) Solving the discrete equation system

A consequent parallelization of the meshfree code should also include the use of a parallel equation solver. That is, not only to solve the discrete equation system in parallel, but also to take over its parallel storage scheme. Otherwise the local discrete equation systems have to be assembled to a global one which also gives rise to the necessity to allocate memory correspondingly at each node. This however, is not desirable as it would limit the problem size which could be modelled. The meshfree code implemented for this thesis utilizes the parallel equation solver package called *Portable Extensible Toolkit for Scientific Computations* (PETSc) which has been developed at the *Argonne National Laboratory* (USA) (Satish *et al.* 2003). It contains various pre-conditioning methods and enables to solve iteratively using different *Krylov subspace methods* as well as directly using the LU-factorization.

12) Visualization of the results

After the discrete equation system has been successfully solved, the locally held portion of the particle parameters $\mathbf{d}^{local}$ is distributed to the other threads so that the particle parameter of all particles P_I can be updated at each thread by

$$u_I^{i+1} = \Delta u_I + u_I^i \,. \tag{B.9}$$

Subsequently, the solution e.g. displacement or stress can be approximated at the particles or the local portion of integration points, depending whether the results are plotted at the particles or the integration points. Finally, all data to be plotted have to be assembled at one *root* process and are written into files, the format of which is according to the requirements of the used visualization software. This thesis makes use of a software called *GID* which has been developed at the *International Center For Numerical Methods In Engineering (CIMNE).*

Appendix C

Iterative stabilization parameter determination algorithm

The basic idea of the iterative stabilization parameter determination is that the parameters β are individually assigned to each numerical integration point and the final configuration ensures that at each integration point a minimum value of essential boundary condition enforcement δ is given. The parameter δ is a constant value applied to the entire problem domain. As already mentioned δ can be related to the displacement convergence norm used for the *Newton-Raphson method*. Furthermore the initial setting of the stabilization parameters β and its successive increase can be approximated by Eq. (4.13). The objective of this algorithm is, however, also to minimize the number of iteration steps needed to compute a suitable distribution for stabilization parameters. The coded algorithm used for this work is found by intensive testing and looks like as follows:

- set the maximum error for all integration points to $\epsilon_{max} = 0$
- loop over all integration points P_I and all their degrees of freedom k with essential boundary conditions applied
  1. compute the current boundary enforcement error $\epsilon = |\, u_k^I - h_k^I \,|$ and set the stabilization parameter β by the following procedure
     (a) if the boundary condition enforcement is not accurate enough $\epsilon > \delta$, then β must be initially set or increased and we first define a factor $m = \frac{\epsilon}{\delta}$
         i. if $m > \beta$ then $\beta = m$
         ii. else if $m < 2.0$ then $\beta = \beta * 2.0$
         iii. else $\beta = \beta * m$
     (b) else if boundary condition enforcement is more accurate then required $\epsilon < \delta * 10$ then gradually reduced the stabilization parameter by $\beta = \beta / 2$
  2. adjust the maximum error for all integration points: if $\epsilon_{max} < \epsilon$ then $\epsilon_{max} = \epsilon$
  3. check, if the boundary enforcement error is for all integration points lower than the given error tolerance $\epsilon_{max} < \delta$ and distinguish two cases

(a) if $\epsilon_{max} > \frac{\delta}{10}$ or it is the first iteration step, then set the parameter of all particles to zero values and proceed with the new stabilization parameter distribution at 2.)

(b) otherwise terminate the iteration procedure

Appendix D

Some definitions and relations of tensor calculus

Some definitions and relations of tensor calculus shall be summarized here. It is assumed that the reader is familiar with basic tensor algebra and analysis. Lower-case greek letters α, β, γ, ... denote scalars, lower-case bold-face roman letters $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, ... vectors and upper-case bold-face roman letter $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, ... second-rank tensors.

The scalar and the cross products of two vector $\mathbf{a}$ and $\mathbf{c}$ are denoted by

$$\mathbf{a} \cdot \mathbf{b} = \alpha \tag{D.1}$$

and

$$\mathbf{a} \times \mathbf{b} = \mathbf{c}\,, \tag{D.2}$$

respectively, and the following relations hold

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a} \tag{D.3}$$

$$\mathbf{a} \cdot (\mathbf{b} + \mathbf{c}) = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c} \tag{D.4}$$

$$(-\mathbf{a}) \times \mathbf{b} = -\mathbf{a} \times \mathbf{b} \tag{D.5}$$

$$\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a} \tag{D.6}$$

$$\alpha\,(\mathbf{a} \times \mathbf{b}) = (\alpha\mathbf{a}) \times \mathbf{b} = \mathbf{a} \times (\alpha\mathbf{b}) \tag{D.7}$$

$$\mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}) = \mathbf{b} \cdot (\mathbf{c} \times \mathbf{a}) = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) \tag{D.8}$$

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\,\mathbf{b} - (\mathbf{a} \cdot \mathbf{b})\,\mathbf{c} \tag{D.9}$$

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) + \mathbf{b} \times (\mathbf{c} \times \mathbf{a}) + \mathbf{c} \times (\mathbf{a} \times \mathbf{b}) = \mathbf{0} \quad \text{(Lagrange identity)} \tag{D.10}$$

$$(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{b} \cdot \mathbf{d})\,(\mathbf{c} \cdot \mathbf{a}) - (\mathbf{a} \cdot \mathbf{d})\,(\mathbf{c} \cdot \mathbf{b})\ . \tag{D.11}$$

The tensor product of two vectors is written as

$$(\mathbf{a}\otimes\mathbf{b}) = \mathbf{A}\,, \tag{D.12}$$

and the following rules are valid

$$\mathbf{A}^T = (\mathbf{a}\otimes\mathbf{b})^T = (\mathbf{b}\otimes\mathbf{a}) \tag{D.13}$$

$$(\mathbf{a}\otimes\mathbf{b})\,\mathbf{u} = (\mathbf{b}\cdot\mathbf{u})\,\mathbf{a} \tag{D.14}$$

$$\mathbf{a}\otimes(\mathbf{b}+\mathbf{c}) = \mathbf{a}\otimes\mathbf{b}+\mathbf{a}\otimes\mathbf{c} \tag{D.15}$$

$$(\alpha\mathbf{a})\otimes\mathbf{b} = \mathbf{a}\otimes\alpha\mathbf{b} = \alpha\,(\mathbf{a}\otimes\mathbf{b}) \tag{D.16}$$

$$\mathbf{T}\,(\mathbf{u}+\mathbf{v}) = \mathbf{T}\mathbf{u}+\mathbf{T}\mathbf{v} \tag{D.17}$$

$$\mathbf{T}\,(\alpha\mathbf{u}) = \alpha\,(\mathbf{T}\mathbf{u}) = (\alpha\mathbf{T})\,\mathbf{u} \tag{D.18}$$

$$(\mathbf{T}+\mathbf{S})\,\mathbf{u} = \mathbf{T}\mathbf{u}+\mathbf{S}\mathbf{u}\,. \tag{D.19}$$

The scalar product of two tensors is denoted by

$$\mathbf{A}:\mathbf{B} = \mathbf{C}\,, \tag{D.20}$$

and the following relations can be found

$$\alpha\,(\mathbf{T}:\mathbf{S}) = (\alpha\mathbf{T}):\mathbf{S} = \mathbf{T}:(\alpha\mathbf{S}) \tag{D.21}$$

$$\mathbf{T}:\mathbf{S} = (\mathbf{a}\otimes\mathbf{b}):(\mathbf{c}\otimes\mathbf{d}) = (\mathbf{a}\cdot\mathbf{c})\,(\mathbf{b}\cdot\mathbf{d}) \tag{D.22}$$

$$\mathbf{T}:(\mathbf{a}\otimes\mathbf{b}) = \mathbf{a}\cdot\mathbf{T}\mathbf{b} = \mathbf{T}^T\mathbf{a}\cdot\mathbf{b} \tag{D.23}$$

$$\mathbf{T}:\mathbf{S} = \mathbf{S}:\mathbf{T} \tag{D.24}$$

$$\mathbf{T}:\mathbf{S} = \mathbf{T}^T:\mathbf{S}^T = \mathbf{T}\mathbf{S}^T:\mathbf{1} \tag{D.25}$$

$$\mathbf{T}:(\mathbf{S}+\mathbf{R}) = \mathbf{T}:\mathbf{S}+\mathbf{T}:\mathbf{R} \tag{D.26}$$

$$\mathbf{R}\mathbf{S}:\mathbf{T} = \mathbf{R}:\mathbf{T}\mathbf{S}^T = \mathbf{T}^T\mathbf{R}:\mathbf{S}^T = \mathbf{T}^T:\mathbf{S}^T\mathbf{R}^T = \mathbf{T}:\mathbf{R}\mathbf{S}\,. \tag{D.27}$$

The tensor product of tensors is denoted by

$$\mathbf{A}\mathbf{B} = \mathbf{C} \tag{D.28}$$

and the following rules are valid

$$\mathbf{T}\mathbf{S} = (\mathbf{a}\otimes\mathbf{b})\,(\mathbf{v}\otimes\mathbf{u}) = (\mathbf{b}\cdot\mathbf{c})\,(\mathbf{a}\otimes\mathbf{d}) \tag{D.29}$$

$$\mathbf{T}\,(\mathbf{u}\otimes\mathbf{v}) = \mathbf{T}\mathbf{u}\otimes\mathbf{v} \tag{D.30}$$

$$(\mathbf{u}\otimes\mathbf{v})\,\mathbf{T} = \mathbf{u}\otimes\mathbf{T}^T\mathbf{v} \tag{D.31}$$

$$(\mathbf{TS})\,\mathbf{R} = \mathbf{T}\,(\mathbf{SR}) \tag{D.32}$$

$$\mathbf{T}\,(\mathbf{R}+\mathbf{S}) = \mathbf{TR}+\mathbf{TS} \tag{D.33}$$

$$(\mathbf{R}+\mathbf{S})\,\mathbf{T} = \mathbf{RT}+\mathbf{ST} \tag{D.34}$$

$$\alpha\,(\mathbf{TS}) = (\alpha\mathbf{T})\,\mathbf{S} = \mathbf{T}\,(\alpha\mathbf{S})\ . \tag{D.35}$$

The tensor product of vector and tensor is written as

$$(\mathbf{u}\times\mathbf{T}) = \mathbf{S}\,, \tag{D.36}$$

and the following relations hold

$$(\mathbf{u}\times\mathbf{T})\,\mathbf{v} = \mathbf{u}\times(\mathbf{Tv}) \tag{D.37}$$

$$(\mathbf{T}\times\mathbf{u})\,\mathbf{v} = (\mathbf{Tv})\times\mathbf{u} \tag{D.38}$$

$$\mathbf{T}:(\mathbf{S}\times\mathbf{v}) = -\mathbf{S}:(\mathbf{T}\times\mathbf{v}) \tag{D.39}$$

$$\mathbf{u}\times\mathbf{T} = -\mathbf{T}\times\mathbf{u} \tag{D.40}$$

$$\mathbf{u}\times(\mathbf{T}+\mathbf{S}) = \mathbf{u}\times\mathbf{T}+\mathbf{u}\times\mathbf{S} \tag{D.41}$$

$$(\mathbf{u}+\mathbf{v})\times\mathbf{T} = \mathbf{u}\times\mathbf{T}+\mathbf{v}\times\mathbf{T} \tag{D.42}$$

$$\alpha\,(\mathbf{u}\times\mathbf{T}) = (\alpha\mathbf{u})\times\mathbf{T} = \mathbf{u}\times(\alpha\mathbf{T}) \tag{D.43}$$

$$(\mathbf{v}\times\mathbf{S}):\mathbf{T} = -\,(\mathbf{v}\times\mathbf{T}):\mathbf{S} \tag{D.44}$$

$$(\mathbf{u}\times\mathbf{I}) = -\,(\mathbf{u}\times\mathbf{I})^T \tag{D.45}$$

$$\mathbf{u}\times(\mathbf{a}\otimes\mathbf{b}) = (\mathbf{u}\times\mathbf{a})\otimes\mathbf{b} \tag{D.46}$$

$$(\mathbf{a}\otimes\mathbf{b})\times\mathbf{u} = (\mathbf{a}\times\mathbf{u})\otimes\mathbf{b} \tag{D.47}$$

Finally, the double contraction of a third-rank $\overset{\mathbf{3}}{\mathcal{K}}$ and second-rank tensor $\mathbf{B}$ is denoted by

$$\overset{\mathbf{3}}{\mathcal{K}}:\mathbf{B} = \mathbf{u}\,. \tag{D.48}$$

Bibliography

[1] Aboudi J. (1991) *A unified micromechanical approach*, Elsevier, Amsterdam

[2] Aifantis E.C. (1984) 'On the microstructural origin of certain inelastic models', *Trans. ASME Journal of Engineering and Materials and Technology*, **106**: 326-330

[3] Aifantis E.C. (1996) 'Higher order gradients and size effects', in *Size-scale effects effects in the failure mechanisms of materials and structures*: 231-242 (edited by Hildebrandt S. and Karcher H.), E & FN Spon, London

[4] Aifantis E.C. (1999) 'Strain gradient interpretation of size effects', *International Journal of Fracture*, **95**: 299-314

[5] Arroyo M. and Ortiz M. (2006) 'Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods', *International Journal for Numerical Methods in Engineering*, **65**: 2167-2202

[6] Arruda E.M. and Boyce M.C. (1993) 'A three-dimensional constitutive model for the large stretch behaviour of rubber elastic materials', *Journal of the Mechanics and Physics of Solids*, **41**: 389-412

[7] Askes H. and Aifantis E.C. (2002) 'Numerical modeling of size effects with gradient elasticity', *International Journal of Fracture*, **117**: 347-358

[8] Atluri S. N. and Zhu T. (1998) 'A new meshless local petrov-galerkin (mlpg) approach to nonlinear problems in computer modeling and simulation', *Computer Modeling and Simulation in Engineering*, **3**: 187-196

[9] Atluri S. N. and Zhu T. (2000a) 'The meshless local petrov-galerkin (mlpg) approach for solving problems in elasto-statics', *Computational Mechanics*, **25**: 169-179

[10] Atluri S.N. and Zhu T. (2000b) 'New concepts in meshless methods' *International Journal for Numerical Methods in Engineering*, **47**: 537-556

[11] Bathe K.J. (1982) 'Finite Element Procedures in Engineering Analysis', Prentice Hall

[12] Bathe K.J. (2001) 'The inf-sup condition and its evaluation for mixed finite element methods', *Computers and Structures*, **49**: 243-252

[13] Beissel S. and Belytschko T. (1996) 'Nodal integration of the element-free Galerkin method', *Computer Methods in Applied Mechanics and Engineering*, **179**:49-74

[14] Belikov V.V., Ivanov V.V., Kontorovich V.K., Korytnik S.A and Semenov A.Y. (1997) 'The non-Sibsonian interpolation: a new method of interpolation of the values of a function on an arbitrary set of points', *Computational Mathematics and Mathematical Physics*, **37**(1): 9-15

[15] Belytschko T., Gu L. and Lu Y.Y. (1994a) 'Fracture and crack growth by element-free Galerkin methods', *Modelling and Simulation in Materials Science and Engineering*, **2**: 519-534

[16] Belytschko T., Lu Y.Y., and Gu L. (1994b) 'Element free galerkin methods', *International Journal for Numerical Methods in Engineering*, **37**: 229-256

[17] Belytschko T. and Tabbara M. (1996a) 'Dynamic fracture using Element-Free Galerkin methods', *International Journal For Numerical Methods In Engineering*, **39**: 923-938

[18] Belytschko T., Krongauz Y., Fleming M., Organ D. and Liu W.K. (1996) 'Smoothing and accelerated computations in the element free galerkin method', *Journal of Computational and Applied Mathematics*, **74**:111-126

[19] Belytschko T., Organ D., and Gerlach C. (2000) 'Elementfree galerkin methods for dynamic fracture in concrete', *Computer Methods in Applied Mechanics and Engineering*, **187**: 385-399

[20] Besdo D. (1974) 'Ein Beitrag zur nichtlinearen Theorie des Cosserat-Kontinuums', *Acta Mechanica*, **20**: 105-131

[21] de Boer R. (1982) *Vektor und Tensorrechnung für Ingenieure*, Springer Verlag, Berlin

[22] de Borst R. and Mühlhaus H.-B. (1992) 'Gradient-dependent plasticity: formulation and algorithm aspects', *International Journal for Numerical Methods in Engineering*, **35**: 521-539

[23] de Borst R. (1993) 'A generalization of J_2-flow theory for polar continua', *Computer Methods in Applied Mechanics and Engineering*, **103**: 347-362

[24] Braun J. and Sambridge M. (1995) 'A numerical method for solving partial differential equations on highly irregular evolving grids', *Nature*, **376**: 655-660

[25] Breitkopf P., Touzot G. and Villon P. (1998) 'Consistency approach and diffuse derivation in element free methods based on moving least squares approximation', *Com. Assist. Mech. Eng. Sci.*, **5**: 479-501

[26] Breitkopf P., Rassineux A., Touzot G. and Villon P. (2000) 'Explicit form and efficient computation of mls shape function and their derivatives', *International Journal for Numerical Methods in Engineering*, **48**: 451-466

[27] Breitkopf P., Rassineux A. and Villon P. (2002) 'An introduction to moving least squares meshfree methods', *Revue Européenne des Eléments Finis*, **11**,(7-8): 825-868

[28] Bufler H. (1985) 'The Biot stresses in nonlinear elasticity and the associated generalized variational principles', *Ingenieur-Archiv*, **55**: 450-462

[29] Cauchy A.L. (1850) 'Mémoire sur les systèms isotropes de points matériels' *Mémoire de l'Acacémie les Sciences, Paris*, **22**

[30] Chen J.-S., Pan C., Wu C.-T., and Liu W.K. (1996) 'Reproducing kernel particle methods for large deformation analysis of nonlinear structures', *Computer Methods in Applied Mechanics and Engineering*, **139**: 195-227

[31] Chen J.-S., Pan C. and Wu C.-T. (1997) 'A large deformation analysis of rubber based on a reproducing kernel particle method', *Computational Mechanics*, **19**: 211-227

[32] Chen J.-S. and Wang H.P. (2000) 'New boundary condition treatments in meshfree computation of contact problems', *Computer Methods in Applied Mechanics and Engineering*, **187**: 441-468

[33] Chen J.-S., Wu C.-T., Yoon S. and You Y. (2001) 'A stabilized conforming nodal integration for Galerkin mesh-free methods', *International Journal for Numerical Methods in Engineering*, **50**: 435-466

[34] Chen J.-S., Wu Y., Wang D. (2004) 'Nonconforming semi-Lagrangian meshfree formulation for problems with excessive particle motion', invited, *Computers And Structures*

[35] Choquet-Bruhat Y., DeWitt-Morette C. and Dillard-Bleick M. (1982) *Analysis, Manifolds and Physics Part I*, North-Holland, Amsterdam

[36] Cosserat E. and F. (1909) *Théorie des corps déformables*, A. Hermann & Fils, Paris

[37] De S. and Bathe K.J. (2000) 'The method of finite spheres', *Computational Mechanics*, **25**: 329-345

[38] De S. and Bathe K.J. (2001) 'Displacement/pressure mixed interpolation in the method of finite spheres', *International Journal for Numerical Methods in Engineering*, **51**: 275-292

[39] Delves L. and Mohamed J. (1985) *Computational Methods for Integral Equations*, *Cambridge University Press*, New York

[40] Dolbow J. and Belytschko T. (1998) 'An introduction to programming the meshless element free Galerkin method', *Archives in Computational Mechanics*, **5**(3): 207-241

[41] Dolbow J. and T. Belytschko (1999) 'Numerical integration of Galerkin weak form in meshfree methods'. *Computational Mechanics*, **23**(3): 219-230

[42] Duarte C.A. and Oden J.T. (1996) 'Hp clouds - a hp meshless method', *Numerical Methods for Partical Differential Equations*, **12**: 673-705

[43] Dubrovin B.A., Fomenko A.T. and Novikov S.P. (1984) *Modern Geometry- Methods and Applications I*, Springer-Verlag, New York

[44] Ericksen J.L. and Truesdell C. (1957) 'Exact theory of stress and strain in rods and shells', *Archive for Rational Mechanics and Analysis*, **1**: 295-323

[45] Eringen A.C. and Suhubi E.S. (1964) 'Nonlinear theory of simple micro-elastic solids', *International Journal of Engineering Science*, **2**: 189-203

[46] Eringen A.C. (1975) *Continuum Physics*, Academic Press, New York

[47] Eringen A.C. and Kafadar C.B. (1976) 'Polar field theories', in *Continuum Physics IV* (Edited by Eringen A.C.), Academic Press, New York

[48] Eringen A.C. (1999) *Microcontinuum field theories I: Foundations and Solids*, Springer, New York

[49] Fleck N.A., Müller G.M., Ashby M.F., Hutchinson J.W. (1994) 'Strain gradient plasticity: theory and experiment', *Acta Metallurgica et Materialia*, **42**: 475-487

[50] Fleck N.A. and Hutchinson J.W. (2001) 'A reformulation of strain gradient plasticity', *Journal of the Mechanics and Physics of Solids*, **49**: 2245-2271

[51] Forest S. (1998) 'Mechanics of generalized continua: construction by homogenization', *Journal de Physique IV*, **8**: 39-48

[52] Forest S., Barbe F. and Cailletaud G. (2000) 'Cosserat modelling of size effects in the mechanical behaviour of polycrystals and multi-phase materials', *International Journal of Solids and Structures*, **37**: 7105-7126

[53] Forest S. and Sievert R. (2003) 'Elastoviscoplastic constitutive frameworks for generalized continua', *Acta Mechanica*, **160**: 71-111

[54] Garrison Jr. W.M. and Moody N.R. (1987) 'Ductile Fracture' *Journal of the Physics and Chemistry of Solids*, **48**: 1035-1074

[55] Gingold R.A. and Monaghan J.J. (1977) 'Smoothed particle hydrodynamics: theory and applications to non-spherical stars' *Monthly Notices of the Royal Astronomical Society*, **181**: 375-389

[56] Gosz J., Liu W.K. (1996) 'Admissible approximation for essential boudary conditions in the reproducing kernel particle method', *Computational Mechanics*, **19**: 120-135

[57] Griebel M. and Schweitzer M.A. (2002) 'A Particle-Partition of Unity Method - Part V': Boundary conditions', in *Geometric Analysis and Nonlinear Partial Differential Equations*: 517-540 (edited by Hildebrandt S. and Karcher H.), Springer

[58] Günther W. (1958) 'Zur Statik und Kinematik des Cosseratschen Kontinuums', *Abhandlungen der Braunschweigischen Wissenschaftlichen Gesellschaft*, **1**: 195-213

[59] Gurson A.L. (1977) 'Continuum theory of ductile rupture by void nucleation and growth. Part I: Yield criteria and flow rules for porous ductile materials', *Journal of Engineering Materials and Technology*, **99**: 2-15

[60] Gutkin M.Y. and Aifantis E.C. (1999) 'Dislocations and disclinations in gradient elasticity', *Physica Status Solidi (b)*, **214**: 245-284

[61] Hestens D. (1990) 'New Foundations for Classical Mechanics', Kluwer Academic Publishers, Dordrecht

[62] Hoek E. and Brown E. T. (1982) *Underground excavations in rock*, The Institution of Mining and Metallurgy, London

[63] Iske A. and Kaeser M. (2005) 'Two-phase flow simulation by AMMoC, an adaptive meshfree method of characteristics', *Computer Modeling In Engineering And Science*, **7**(2): 133-148

[64] Jelenic G. and Crisfield M.A. (1999) 'Geometrically exact 3D beam theory: implementation of a strain-invariant finite element for static and dynamics', *Computer Methods in Applied Mechanics and Engineering*, **171**: 141-171

[65] Jun S., Liu W.K. and Belytschko T. (1998) 'Explicit reproducing kernel particle methods for large deformations problems', *Journal of Numerical Methods in Engineering*, **41**: 137-166

[66] Karypis G. and Kumar V. (1997) 'A coarse-grain parallel multilevel *k*-way partitioning algorithm' *Proceedings of the Eigth SIAM Conference on Parallel Processing For Scientific Computing*

[67] Karypis G. and Kumar V. (1998) 'Multilevel k-way Partitioning Scheme for Irregular Graphs' *Journal of Parallel and Distributed Computing*, **48**(1): 96-129

[68] Koiter W.T. (1964) 'Couple stresses in the theory of elasticity', *Proceedings of the Royal Netherlands Academy of Arts and Sciences*, **67**: 17-44

[69] Kröner E. (1958) 'Kontinuumstheorie der Versetzungen und Eigenspannungen', *Ergebnisse der Angewandten Mathematik*, **3**: 1-179

[70] Krongauz Y. and Belytschko T. (1996) 'Enforcement of essential boundary conditions in meshless approximations using finite elements', *Computer methods in applied mechanics and engineering*, **131**: 133-145

[71] Lakes R.S. (1986) 'Experimental microelasticity of two porous solids', *International Journal of Solids and Structures*, **22**(1): 55-63

[72] Lam D.C.C., Yang F., Chong A.C.M., Wang J. and Tong P. (2003) 'Experiments and theory in strain gradient elasticity', *Journal of the Mechanics and Physics of Solids*, **51**: 1477-1508

[73] Lancaster P. and Salkauskas K. (1981) 'Surface generated by moving least square methods', *Mathematics of Computations*, **37**(155): 141-158

[74] Li S. and Liu W.K. (1998) 'Reproducing kernel hierarchical partition of unity part I', *Journal of Numerical Methods in Engineering*, **45**: 251-288

[75] Li S. and Liu W.K. (2000) 'Numerical simulations of large deformations of thin shell structures using meshfree methods', *Computational Mechanics*, **25**: 102-116

[76] Li S., Liu W.K., Quian D., Gudrun P. and Rosakis R. (2001) 'Dynamic shear band propagation and micro-structure of adiabatic shear band', *Computer Methods In Applied Mechanics And Engineering*, **191**: 73-92

[77] Lippmann H. (1969) 'Eine Cosserat-Theorie des plastischen Fließens', *Acta Mechanica*, **8**: 255-284

[78] Lippmann H. (1995) 'Cosserat plasticity and plastic spin', *Applied mechanics reviews*, **48**: 753-762

[79] Liszka T., Duarte C.A.M., and Tworzydlo W.W. (1996) 'Hp-meshless cloud method', *Computer Methods in Applied Mechanics and Engineering*, **139**: 263-288

[80] Liu G.R. and Chen X.L. (2001) 'A mesh-free method for static and free vibration analysis of thin plates of complicated shape', *Journal of Sound and Vibration*, **241**(5): 839-855

[81] Liu W.K. and Chen Y. (1995) 'Wavelet and multiple scale reproducing kernel methods', *Journal of Numerical Methods in Fluids*, **21**: 901-931

[82] Liu W.K., Chen Y., Uras R.A. and Chang C.T. (1996) 'Generalized multiple scale reproducing kernel particle methods', *Computer Methods in Applied Mechanics and Engineering*, **139**: 733-752

[83] Liu W.K., Jun S. and Belytschko T. (1995) 'Reproducing kernel particle methods', *International Journal for Numerical Methods in Fluids*, **11**: 83-95

[84] Liu W.K., Li S. and Belytschko T. (1997) 'Moving least square reproducing kernel method part I: Methodology and convergence', *Computer Methods in Applied Mechanics and Engineering*, **143**: 422-433

[85] Lucy L.B. (1977) 'A numerical approach to the testing of the fission hypothesis', *Astrophysical Journal*, **82**: 1013

[86] Lu Y.Y., Belytschko T. and Gu L. (1994) 'A new implementation of the element free Galerkin method', *Computer Methods in Applied Mechanics and Engineering*, **113**: 397-414

[87] Marsden J. E. and Hughes T.J.R (1983) 'Mathematical foundations of elasticity', Prentice Hall

[88] Mase G.T. and Mase G.E. (1999) *Continuum mechanics for engineers*, CRC Press; Boca Raton, London, New York, Washington D.C.

[89] Melenk J.M. and Babuska I. (1997) 'The partition of unity method: Basic theory and applications' *International Journal for Numerical Methods in Engineering*, **40**: 727-758

[90] Mindlin R.D. (1964) 'Micro-structure in linear elasticity', *Archive for Rational Mechanics and Analysis*, **16**: 51-78

[91] Monaghan J.J. (1985) 'Particle methods for hydrodynamics' *Computer Physics Report*, **3**: 71-124

[92] Mühlhaus H-B. and Vardoulakis I. (1987) 'The thickness of shear bands in granular materials', *Géotechnique*, **37**: 271-283

[93] Mühlhaus H.-B. (1989) 'Application of Cosserat theory in numerical solution of limit load problems', *Ingenieur-Archiv*, **59**: 124-137

[94] Mühlhaus H.-B. and Aifantis E.C. (1991) 'A variational principle for gradient plasticity', *International Journal of Solids and Structures*, **28**(7): 845-857

[95] Nayroles B., Touzot G. and Villon P. (1992) 'Generalizing the finite element method: Diffuse approximation and diffuse elements', *Computational Mechanics*, **10**: 307-318

[96] Nitsche J. (1970-1971) 'Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind', *Abhandlungen aus den Mathematischen Seminaren der Universität Hamburg*, **36**: 9-15

[97] Nye J.F. (1953) 'Some geometrical relations in dislocated crystals', *Acta Metallurgica et Materialia*, **1**: 153-162

[98] Oden J.T., Duarte C.A.M., and Zienkiewicz O.C. (1998) 'A new cloud-based hp finite element method', *Computer Methods in Applied Mechanics and Engineering*, **153**: 117-126

[99] Organ D., Fleming M., Terry T. and Belytschko T. (1996) 'Continuous meshless approximations for nonconvex bodies by diffraction and transparency' *Computational Mechanics* **18**(3): 225 - 235

[100] Pamin J., Askes H. and de Borst R. (2003) 'Two gradient plasticity theories discretized with the element-free Galerkin method', *Computer Methods in Applied Mechanics and Engineering*, **192**: 2377-2403

[101] Powell J. D. (1969) 'A method for non-linear constraint in optimization problems', in *Optimization*, Fletcher R. (ed.), Academic Press, London, 283-298

[102] Randles P.W., Libersky L.D. and Petschek A.G. (1999) 'On neighbours, derivatives and viscosity in particle codes', *Proceeding of ECCM Conference*, Munich, Germany, 31 August September 1999

[103] Rousselier G. (1987) 'Ductile Fracture Models and Their Potential in Local Approach of Fracture', *Nuclear Engineering and Design*, **105**: 97-111

[104] Schäfer H. (1967) 'Das Cosserat-Kontinuum', *Zeitschrift für angewandte Mathematik und Mechanik*, **47**: 485-498

[105] Sansour C. and Bufler H. (1992) 'An exact finite rotation shell theory, its mixed variational formulation, and its finite element implementation', *International Journal for Numerical Methods in Engineering*, **34**: 73-115

[106] Sansour C. and Bednarczyk H. (1995) 'The Cosserat surface as shell model, theory and finite-element formulation', *Computer Methods in Applied Mechanics and Engineering*, **120**: 1-32

[107] Sansour C. (1998a) 'A theory of the elastic-viscoplastic Cosserat continuum', *Archive of Mechanics*, **50**: 577-597

[108] Sansour C. (1998b) 'A unified concept of elastic-viscoplastic Cosserat and micromorphic continua', *Journal de Physique IV Proceedings*, **8**: 341-348

[109] Sansour C., Feih S. and Wagner W. (2003a) 'On the performance of enhanced strain finite elements in large strain deformations of elastic shells', *International Journal for Computer-Aided Engineering and Software*, **20**(7): 875-895

[110] Sansour C. and Wagner W. (2003b) 'Multiplicative Updating of the Rotation Tensor in the Finite Element Analysis of Rods and Shells - A Path Independent Approach', *Computational Mechanics*, **31**: 153-162

[111] Satish B., Gropp W.D., McInnes L.C. and Smith B.F. (2003) *PETSc user manual - revision 2.1.5*, Argonne National Laboratory, USA

[112] Shepard D. (1968) 'A two-dimensional interpolation function for irregular spaced points', *Proceedings of A.C.M. National Conference*, 517-524

[113] Sibson R.A. (1980) 'A vector identity for Dirichlet tesselation', *Mathematical Proceedings of the Cambridge Philosophical Society*, **87**: 151-155

[114] Sidoroff F. (1975) ' Microstructure and plasticity', *Mechanics Research Communications*, **2**(2): 73-77

[115] Simo J.C. and Vu-Quoc L. (1986) 'A three-dimensional finite-strain rod model. Part II: Geometric and computational aspects.', *Computer Methods in Applied Mechanics and Engineering*, **58**: 79-116

[116] Simo J.C., Tarnow N. and Doblare M. (1995) 'Nonlinear dynamics of three-dimensional rods: exact energy and momentum conservation algorithms', *International Journal for Numerical Methods in Engineering*, **58**: 79-116

[117] Sievert R. (1992) 'Eine Systematik für elastisch-plastische Stoffgleichungen', Disseratation, Technische Universität Berlin

[118] Skatulla S. and Sansour C. (2005) 'On meshfree computations of shells', Proceedings, Third MIT Conference, Boston, USA

[119] Sladek J. and Sladek V. (2003) 'Application of local boundary integral equation method into micropolar elasticity', *Engineering Analysis with Boundary Elements*, **27**: 81-90

[120] Spencer A.J.M (1980) 'Continuum mechanics', Longman Group, London

[121] Steinmann P. (1994) 'A micropolar theory of finite deformation and finite rotation multiplicative elastoplasticity', *International Journal for Solids and Structures*, **31**(8): 1063-1084

[122] Stenberg R. (1995) 'On some techniques for approximating boundary conditions in the Finite Element Method', *Journal of Computational and Applied Mathematics*, **63**:139-148

[123] Stenberg R. (1998) 'Mortaring by a method by J.A. Nitsche', in *Computational Mechanics, New Trends and Applications* (edited by S. Idelsohn, E. Onate and E. Dvorkin), CIMNE, Barcelona, Spain

[124] Sukumar N., Moran B. and Belytschko T. (1998) 'The natural element method in solid mechanics', *International Journal for Numerical Methods in Engineering*, **43**(5): 839-887

[125] Sukumar N., Moran B., Semenov Y. and Belytschko T. (2001) 'Natural neighbour Galerkin methods', *International Journal for Numerical Methods in Engineering*, **50**: 1-27

[126] Sun D.-Z., Sester M. and Schmitt W., (1997) 'Development and application of micromechanical material models for ductile fracture and creep damage', *International Journal for Fracture*, **86**: 75-90

[127] Tang Z., Shen S. and Atluri S.N. (2003) 'Analysis of materials with strain gradient effects: A Meshless Local Petrov-Galerkin (MLPG) approach, with nodal displacements only', *Computer Modeling in Engineering and Science*, **4**(1): 177-196

[128] Toupin R.A. (1962) 'Elastic materials with couple stresses', *Archive for Rational Mechanics and Analysis*, **11**: 385-414

[129] Traversoni L. (1994) 'Natural neighbour finite elements', In *International Conference on Hydraulic Engineering Software, Hydrosoft Proceedings*, **2**: 291-297

[130] Triantafyllidis N. and Aifantis E.C. (1986) 'A gradient approach to localization of deformation. I. Hyperelastic materials', *Journal of Elasticity*, **16**: 225-237

[131] Truesdell C. and Noll W. (1965) 'The nonlinear field theories of mechanics', in *Handbuch der Physik*, Vol. III/3 (Edited by Flügge S.), Springer, Berlin

[132] Tvergaard V. and Needleman A. (1995) 'Effects of nonlocal damage in porous plastic solids', *International Journal of Solids and Structures*, **32**: 1063-1077

[133] Vardoulakis I., Exakaktylos G. and Kourkoulis S.K. (1998) 'Bending of marble with intrinsic length scales: A gradient theory with surface energy and size effects', *Journal de Physique IV France*, **8**: 399-406

[134] Ventura G. (2002) 'An augmented Lagrangian approach to essential boundary conditions in meshless methods' *International Journal for Numerical methods in Engineering*, **53**: 825-842

[135] Voigt W. (1887) 'Theoretische Studien über die Elastizitätverhältnisse der Kristalle', *Abhandlungen der Gesellschaft der Wissenschaften zu Göttingen, mathematisch-physikalische Klasse*, **34**: 47 *et seqq.*

[136] Wagner G.J. and Liu W.K. (2000) 'Application of essential boundary conditions in mesh-free methods: a corrected collocation method', *International Journal for Numerical Methods in Engineering*, **47**(2): 1367-1379

[137] Walsh S.D.C. and Tordesillas A. (2004) 'A thermomechanical approach to the development of micropolar constitutive models of granular media', *Acta Mechanica*, **167**: 145-169

[138] Washizu K. (1975) *Variational methods in elasticity and plasticity*, Pergamon Press, second edition, Bath, Great Britain

[139] Wei Y., Hutchinson J.W. (1997) 'Steady-state crack growth and work of fracture for solids characterized by strain gradient plasticity', *Journal of the Mechanics and Physics of Solids*, **45**: 1253-1273

[140] Williams T.O. (2005) 'A three-dimensional, higher-order, elasticity-based micromechanics model', *International Journal of Solids and Structures*, **42**: 907-1007

[141] Wu C.-T., Chen J.-S., Chi L., Huck F. (2001) 'A meshfree method for geotechnical materials', *Journal Of Engineering Mechanics*, **127**: 440-449

[142] Xia Z.C. and Hutchinson J.W. (1996) 'Crack tip fields in strain gradient plasticity', *Journal of the Mechanics and Physics of Solids*, **44**(10): 1621-1648

[143] Xiong S., Liu W. K., Cao J., Li C.S., Rodrigues J.M.C. and Martins P.A.F. (2005) 'Simulation of bulk metal forming processes using the reproducing kernel particle Method', *Computers And Structures*, **83**: 574-587

[144] Yang J.F.C and Lakes R.S (1981) 'Transient study of couple stress effects in compact bone: Torsion', *Journal of Biomechanical Engineering*, **103**: 275-279

[145] Zervos A., Papanastasiou P. and Vardoulakis I. (2001) 'Modelling of localization and scale effect in thick-walled cylinders with gradient elastoplasticity', *International Journal of Solids and Structures*, **38**: 5081-5095

[146] Zhou W., Zhao J., Liu Y. and Yang Q. (2002) 'Simulation of localization failure with strain-gradient-enhanced damage mechanics', *International Journal for Numerical and Analytical Methods in Geomechanics*, **26**: 793-813

[147] Zhu H.T., Zbib H.M. and Aifantis E.C. (1997) 'Strain gradients and continuum modelling of size effect in metal matrix composites', *Acta Mechanica*, **121**: 165-176

[148] Zhu T. (1999) 'A new meshless regular local boundary integral equation (mrlbie) approach', *International Journal for Numerical Methods in Engineering*, **46**: 1237-1252

[149] Zhu Y. and Fox P.J. (2001) 'Smooth particle hydrodynamics model for diffusion through porous media', *Transport In Porous Media*, **43**: 441-471

[150] Zhu T., Zhang J. and Atluri S.N. (1999) 'A meshless numerical method based on the local boundary integral equation (lbie) to solve linear and non-linear boundary value problems', *Engineering Analysis with Boundary Elements*, **23**: 375-389