Journal of Applied Mechanics Vol. 3, pp. 167-175 (August 2000) JSCE

Linear Structural Analysis Using Cover Least Square Approximation

被覆最小自乗近似法を用いた線形構造解析

Chuanrong JIN[1], Katsuyuki SUZUKI[2] and Hideomi OHTSUBO[1]
金传荣[1]，鈴木克幸[2]，大坪英臣[1]

[1]School of Engineering, University of Tokyo (7-3-1 Honggo, Bunkyo, Tokyo 113-8656, Japan)
[2]School of Frontier Sciences, University of Tokyo (7-3-1 Honggo, Bunkyo, Tokyo 113-8656, Japan)

The voxel analysis method and the meshless method have been proposed to relieve the formidable task of mesh generation of traditional FEM. The model generation of the voxel analysis is very simple, but how to get reasonable accuracy and avoid the sharp increase of degrees of freedom is a problem of the voxel analysis. On the other hand, the meshless methods can avoid the mesh generation completely. However, some difficulties limit the freedom of meshless methods very much. One of these difficulties is how to easily guarantee the linear independence of approximation functions. Recently a new meshless approximation method, Cover Least Square Approximation, has been proposed, by which the linear independence conditions can be simply satisfied. In this paper, it is implemented for linear structural analysis by distributing the covers using multi-resolution voxel data.

Key Words: CLSA, Meshless, FCM, FEM, Structural Analysis

1. Introduction

Mesh generation for the traditional numeric analysis methods such as FEM and FDM is in general very labor-intensive, time-consuming and experience-dependent. To overcome this bottleneck of CAE, numerous research activities have been devoted into the development of automatic mesh generation techniques. As different approaches, the voxel analysis and the so-called meshless method have been proposed and attracted keen interests of many researchers.

The model generation of the voxel analysis is so simple and effective that it is possible to mesh any complex structure that can be manufactured. However, because all the elements are uniform, any local refining requirement causes a global refining. When small voxels are necessary to improve the accuracy in a local area, a numerous number of elements and degrees of freedom are required. How to get reasonable accuracy and avoid the sharp increase of degrees of freedom is a problem of the voxel analysis.

On the other hand, establishing the approximation on some scatted particles in the analysis domain without the needs of traditional mesh, the meshless methods avoid the mesh generation completely. Several versions of the meshless method have been proposed; among them are Generalized Finite Difference Method[1], Smoothed Particle Hydrodynamics[2], Diffuse Element Method[3], Multiquadrics[4][5], Element-Free Galerkin Method[6], Wavelet Galerkin Method[7], hp-cloud method[8], Reproducing Kernel Particle Method[9] and Partition of Unity FEM[10]. An overview can be found in reference 11). However, there yet exist some problems on the way of improving the practicability of the meshless method. Especially some difficulties of implementation still remain[11]. At the end of reference 10), three open questions are addressed. Among them, the choice of a basis of the approximation space is discussed as a major problem. If the basis of approximation space is not selected properly the shape functions will be linearly dependent. As pointed out in reference 8), behind all the meshless methods there is an underlying approximation technique. As known, the linear independence of the shape functions is the most important for any approximation. Otherwise other discussions such as convergence will lose the foundation. In the recent reports about meshless method, more profoundly is mathematical background established more attention is paid to this point.

About the linear independence of meshless approximation, some researches have been presented recently. Babuska et al.[10] have suggested a kind of weight function for 1D problem of Partition of Unity FEM, but the extension to multi-dimension does not

seem easy. We[12] have proposed two kinds of sufficient conditions to construct weight functions for Finite Cover Method (FCM), which have been successfully utilized for 2D and 3D FCM with uniform cover distribution based on voxel data. However, if the cover distribution is not uniform, the weight function is too difficult to be constructed by the proposed conditions. Liu et al.[9] have pointed out that not all particle distributions can be used in numerical computation, and the concept of "admissible particle distribution" is introduced. It implies that the linear independence is guaranteed by carefully dealing with the particle distribution, but how to implement the admissible particle distribution is not shown. A more mathematical discussion has been given in the hp-cloud method[8]. In the hp-cloud method, Oden et al. have introduced a new family of functions to construct the approximation, and the linear independence of the constructed functions has been proven mathematically. However, it is regretted that their proof is not sufficient, and the linear independence of their proposed functions cannot be guaranteed.

The purpose of meshless methods is to ameliorate the formidable task of mesh generation of traditional FEM. If the nodes have to be distributed with extreme attention to linear independence, the freedom of meshless methods will be limited very much. Especially if the method how to guarantee the linear independence is not clear, the meshless methods will not be able to become practicable.

Recently a new meshless approximation method is proposed, named as Cover Least Square Approximation (CLSA)[14][15]. By the CLSA, not only the approximation accuracy can be controlled easily by locally justifying the cover order or the density of cover distribution but also the linear independent conditions can be satisfied conveniently. In this paper, the CLSA will be implemented for linear structural analysis. In order to apply this method practicably, the multi-resolution voxel data will be used to distribute the covers.

2. Basic Formulations of CLSA

The CLSA is established on the base of Finite Cover System. In order to make the model generation convenient, the mathematical approximation is separated from the physical field and cover is classified into mathematical cover and physical cover in the Finite Cover System[15][16].

Let $u(x)$ be a sufficiently smooth function (for example, $u(x)\in C^0(\Omega)$ at least.), which is defined on a simply connected open set Ω.

Let $\{\Omega_i\}$, $i=1,2,\cdots,NC$ be open covers of Ω satisfying the following conditions,

(a)

$$\exists M\in N, \forall x\in\Omega, \quad 1\le \textit{Number of elements in } \{i \mid x\in\Omega_i\}\le M \tag{1}$$

(b) $$\forall\Omega_i, \dim(\Omega_i\cap\Omega)=\dim(\Omega) \tag{2}$$

NC is the total number of covers. Let local approximation function space $V_i(\Omega_i)$ be given subordinate to Ω_i, with a complete basis of $\{\phi_{ij}\}_{j=0}^{n_i}$. On each cover, $u(x)$ can be approximated by a series, called cover function, locally as

$$u(x)\cong\Phi_i(x,O_i)=\sum_{j=0}^{n_i}d_{ij}\phi_{ij}(x-O_i)=\phi_i\mathbf{d}_i,\quad x\in\Omega_i \tag{3}$$

where O_i is the origin of the i th cover, and

$$\phi_i=\{\phi_{i0},\phi_{i1},\phi_{i2},\cdots,\phi_{in_i}\} \tag{4}$$

$$\mathbf{d}_i=\{d_{i0},d_{i1},d_{i2},\cdots,d_{in_i}\}^T \tag{5}$$

In the case of that it is a polynomial series, it becomes a Taylor expansion of $u(x)$. When the series is an infinite one, the cover function becomes equal to $u(x)$.

$$u(x)=\lim_{n_i\to\infty}\Phi_i(x,O_i),\quad x\in\Omega_i \tag{6}$$

Now let us consider certain point $\bar{x}\in\Omega$. According to $\bar{x}$, $u(x)$ can also be expended as a series,

$$u(x)\cong u^l(x,\bar{x})=\sum_{j=0}^{n}a_j(\bar{x})\phi_j(x-\bar{x})=\phi\mathbf{a} \tag{7}$$

where

$$\phi=\{\phi_0,\phi_1,\phi_2,\cdots,\phi_n\} \tag{8}$$

$$\mathbf{a}=\{a_0,a_1,a_2,\cdots,a_n\}^T \tag{9}$$

This series is called here local expansion function.

Since the series is finite, there exists a residual

$$e(x,\bar{x}) = u(x) - u^l(x,\bar{x}) \tag{10}$$

On each cover, substituting the cover function for $u(x)$, the residual is expressed as

$$e_i(x,\bar{x}) = \Phi_i(x,O_i) - u^l(x,\bar{x}) = \phi_i \mathbf{d}_i - \phi \mathbf{a} \tag{11}$$

A functional associated with these residuals is defined as

$$J(a(\bar{x})) = \sum_i w_i(\bar{x}) \int_{\Omega_i} \omega_i(x) e_i^{\,2}(x,\bar{x}) dx \tag{12}$$

$w_i(\bar{x})$ is called influence degree function or weight function, which limits the number of covers involved in the approximation around $\bar{x}$.

$$supp\, w_i = closure(\Omega_i) \tag{13}$$

where $supp\, w_i$ is the support of weight function w_i.

$\omega_i(x)$ is called localization factor function, which localizes the cover approximation.

$$supp\, \omega_i \subseteq closure(\Omega_i) \tag{14}$$

where $supp\, \omega_i$ is the support of localization factor function ω_i.

The weight function $w_i(\bar{x})$ defines that how much the local cover approximation on the i th cover influences the approximation at $\bar{x}$. The bigger is $w_i(\bar{x})$, the greater influence does the i th cover have. $w_i(\bar{x}) = 0$ means that $\bar{x}$ is outside the i th cover and there is no influence of the i th cover to the approximation at $\bar{x}$. Whereas the localization factor function $\omega_i(x)$ localizes the evaluation domain of the residual square. In the following implement, $\omega_i(x)$ is set identical to 1 in its support and its support is set as half of Ω_i; the weight function is defined as trilinear functionin 3D.

Since (13) and (14), the functional (12) can be extended over the whole domain

$$J(a(\bar{x})) = \int_{\Omega} \sum_i w_i(\bar{x}) \omega_i(x) e_i^{\,2}(x,\bar{x}) dx \tag{15}$$

By minimizing the quadratic functional $J(a(\bar{x}))$, we can obtain

$$\mathbf{P}(\bar{x})\mathbf{a}(\bar{x}) = \mathbf{Q}(\bar{x})\mathbf{d} \tag{16}$$

where

$$\mathbf{P}(\bar{x}) = [P_{lm}] \tag{17}$$

$$P_{lm} = \int_{\Omega} \sum_i (\phi_l(x,\bar{x}) w_i(\bar{x}) \omega_i(x) \phi_m(x,\bar{x})) dx \tag{18}$$

$$\mathbf{Q}(\bar{x}) = [\mathbf{Q}_1(\bar{x}), \mathbf{Q}_2(\bar{x}), \cdots, \mathbf{Q}_{NC}(\bar{x})] \tag{19}$$

$$\mathbf{Q}_i(\bar{x}) = [Q_{lk_i}] \tag{20}$$

where

$$Q_{lk_i} = \int_{\Omega} \phi_l(x,\bar{x}) w_i(\bar{x}) \omega_i(x) \phi_{ik_i}(x,O_i) dy \tag{21}$$

$$k_i = 0,1,\cdots,n_i$$

$$\mathbf{d} = [\mathbf{d}_1^{\,T}, \mathbf{d}_2^{\,T}, \cdots, \mathbf{d}_{NC}^{\,T}]^T \tag{22}$$

Since $\omega_i(x) \geq 0$, $w_i(\bar{x}) \geq 0$, and ϕ_i, $i = 0,1,\cdots,n$, are linearly independent, the determinant of $\mathbf{P}(\bar{x})$ is always positive and $\mathbf{P}(\bar{x})$ is always invertible; therefore, the unknown vector $\mathbf{a}(\bar{x})$ is uniquely determined,

$$\mathbf{a}(\bar{x}) = \mathbf{P}^{-1}(\bar{x})\mathbf{Q}(\bar{x})\mathbf{d} \tag{23}$$

Substituting (23) into (7), the approximation around $\bar{x}$ can be written as

$$u(x) \cong u^l(x,\bar{x}) = \phi \mathbf{P}^{-1}(\bar{x})\mathbf{Q}(\bar{x})\mathbf{d} \tag{24}$$

Now the global approximation can be defined as

$$\hat{u}(x) = \lim_{\bar{x} \to x} u^l(x,\bar{x}) = \phi(0)\mathbf{P}^{-1}(x)\mathbf{Q}(x)\mathbf{d} \tag{25}$$

Note that x in (18) and (21) is a dummy variable, (18) and (21) can be rewritten as

$$P_{lm}(x) = \int_{\Omega} \sum_{i} (\phi_l(y,x) w_i(x) \omega_i(y) \phi_m(y,x)) dy \quad (26)$$

$$Q_{lk_i} = \int_{\Omega} \phi_l(y,x) w_i(x) \omega_i(y) \phi_{ik_i}(y,O_i) dy \quad (27)$$

$$k_i = 0,1,\cdots,n_i$$

(25) can be expressed as a shape function expression.

$$\hat{u}(x) = \phi(0)\mathbf{P}^{-1}(x)\mathbf{Q}(x)\mathbf{d} = \mathbf{N}\mathbf{d} = \sum_{i,j} N_{ij} d_{ij} \quad (28)$$

where

$$\mathbf{N}(x) = \phi(0)\mathbf{P}^{-1}(x)\mathbf{Q}(x) \quad (29)$$

The partial derivatives of shape functions can be obtained as follows.

$$\begin{aligned} \mathbf{N}_{,s}(x) &= \phi(0)(\mathbf{P}^{-1}{}_{,s}\mathbf{Q} + \mathbf{P}^{-1}\mathbf{Q}_{,s}) \\ &= \phi(0)(-\mathbf{P}^{-1}\mathbf{P}_{,s}\mathbf{P}^{-1}\mathbf{Q} + \mathbf{P}^{-1}\mathbf{Q}_{,s}) \end{aligned} \quad (30)$$

The index following a comma means a partial derivative.

$$\mathbf{P}_{,s} = [P_{lm,s}] \quad (31)$$

$$\begin{aligned} P_{lm,s} = \int_{\Omega} \sum_{i} & (w_{i,s}(x)\phi_l(y,x)\omega_i(y)\phi_m(y,x) \\ & + w_i(x)\omega_i(y)(\phi_l(y,x)\phi_{m,s}(y,x) + \phi_{l,s}(y,x)\phi_m(y,x)))dy \end{aligned} \quad (32)$$

$$\mathbf{Q}_{,s}(x) = [\mathbf{Q}_{1,s}(x), \mathbf{Q}_{2,s}(x), \cdots, \mathbf{Q}_{NC,s}(x)] \quad (33)$$

$$\mathbf{Q}_{i,s} = [Q_{lk_i,s}] \quad (34)$$

$$Q_{lk_i,s} = \int_{\Omega} (\phi_l w_{i,s}(x) + \phi_{l,s} w_i(x)) \omega_i(y) \phi_{ik_i} dy \quad (35)$$

3. Multi-resolution Voxel Data Based Cover Distribution

The covers are distributed based on voxel data and the density of cover distribution is controlled by voxel size. A voxel here is a rectangular parallelepiped in 3D or a rectangle in 2D. The multi-resolution voxel data consists of groups of voxels, and each group of voxels has the same size.

The multi-resolution voxel data can be generated through the following process. The analysis domain Ω is given as figure 1(a), as an example in 2D.

(1) Give a rectangular parallelepiped in 3D or a rectangle in 2D including the analysis domain Ω completely as Figure 1(b), for example a circumscribed one. This rectangular parallelepiped or rectangle is called 0-level resolution voxel.

(2) Divide the 0-level resolution voxel into some smaller uniform rectangular parallelepipeds or rectangles by regular grid as Figure 1(c). Leave only those that have common field with the analysis domain Ω as Figure 1(d). These left rectangular parallelepipeds or rectangles are called 1-level resolution voxels.

(3) If necessary, certain field covered by 1-level voxels can be divided into smaller uniform rectangular parallelepipeds or rectangles by regular grid as Figure 1(e), and leave those that have common field with the analysis domain Ω as Figure 1(f). These smaller left rectangular parallelepipeds or rectangles are called 2-level resolution voxels.

(4) Recurringly, n-level resolution voxels can be created. It should be noted that the size of i-level resolution voxel must be integral times of the size of the i+1-level resolution voxel.

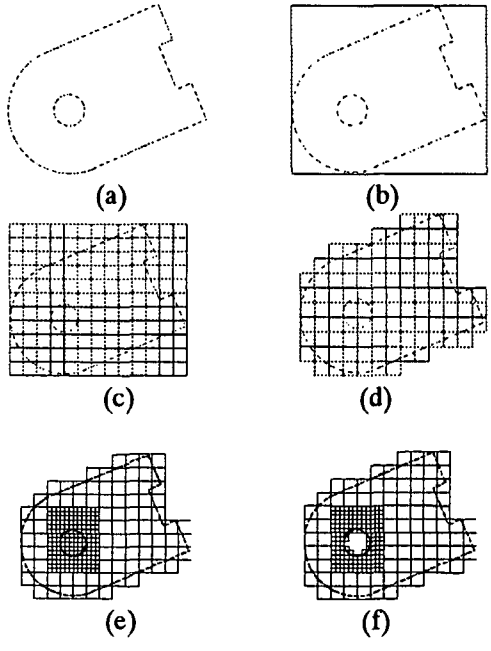


Figure 1 Multi-resolution voxel data

After the multi-resolution voxel data is generated, the covers can be distributed on it by the following way. For simplicity, formulae are given only for 3D.

An example of 2D is given asshown in Figure 2.

Corresponding to each voxel vortex, one mathematical cover is allocated. The cover origin $O_i(x_i, y_i, z_i)$ is allocated at the vortex. Each voxel is defined as an element, and all the cover origins on its sides are said belonging to it. The cover field is determined as a rectangular parallelepiped

$$\Omega_i = \{(x,y,z) \mid -\rho_{mx} < x - x_i < \rho_{px}, -\rho_{my} < y - y_i < \rho_{py}, -\rho_{mz} < z - z_i < \rho_{pz}, \rho_{mx}, \rho_{px}, \rho_{my}, \rho_{py}, \rho_{mz}, \rho_{pz} \geq 0\} \quad (36)$$

ρ_{mx}, ρ_{px}, ρ_{my}, ρ_{py}, ρ_{mz} and ρ_{pz} are determined by the following rules, in which S_i is a set consisting of all the cover origins that belongs to the same element with $O_i(x_i, y_i, z_i)$.

$$\rho_{mx} = \begin{cases} \min\limits_{y_k = y_i, z_k = z_i, x_k < x_i} (x_i - x_k) & if \exists O_k(x_k, y_k, z_k) \in S_i, y_k = y_i, z_k = z_i, x_k < x_i \\ -\min\limits_{S_i}(x_k - x_i) & otherwise \end{cases} \quad (37)$$

$$\rho_{px} = \begin{cases} \min\limits_{y_k = y_i, z_k = z_i, x_k > x_i} (x_k - x_i) & if \exists O_k(x_k, y_k, z_k) \in S_k, y_k = y_i, z_k = z_i, x_k > x_i \\ \max\limits_{S_i}(x_k - x_i) & otherwise \end{cases} \quad (38)$$

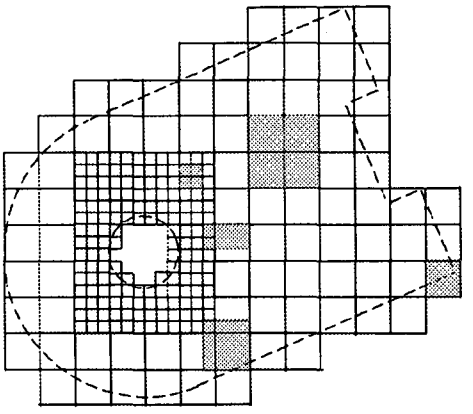
$$\rho_{my} = \begin{cases} \min\limits_{x_k = x_i, z_k = z_i, y_k < y_i} (y_i - y_k) & if \exists O_k(x_k, y_k, z_k) \in S_i, x_k = x_i, z_k = z_i, y_k < y_i \\ -\min\limits_{S_i}(y_k - y_i) & otherwise \end{cases} \quad (39)$$

$$\rho_{py} = \begin{cases} \min\limits_{x_k = x_i, z_k = z_i, y_k > y_i} (y_k - y_i) & if \exists O_k(x_k, y_k, z_k) \in S_i, x_k = x_i, z_k = z_i, y_k > y_i \\ \max\limits_{S_i}(y_k - y_i) & otherwise \end{cases} \quad (40)$$

$$\rho_{mz} = \begin{cases} \min\limits_{x_k = x_i, y_k = y_i, z_k < z_i} (z_i - z_k) & if \exists O_k(x_k, y_k, z_k) \in S_i, x_k = x_i, y_k = y_i, z_k < z_i \\ -\min\limits_{S_i}(z_k - z_i) & otherwise \end{cases} \quad (41)$$

$$\rho_{pz} = \begin{cases} \min\limits_{x_k = x_i, y_k = y_i, z_k > z_i} (z_k - z_i) & if \exists O_k(x_k, y_k, z_k) \in S_i, x_k = x_i, y_k = y_i, z_k > z_i \\ \max\limits_{S_i}(z_k - z_i) & otherwise \end{cases} \quad (42)$$

If the mathematical cover field Ω_i is divided into K_i sub-field Ω_{ik} by the boundary and discontinuous interface of analysis domain, corresponding to K_i sub-field Ω_{ik}, K_i physical covers with the same locations as the mathematical cover Ω_i are allocated.



Mathematical Cover field

Boundary of analysis domain

Figure 2 Cover distribution

4. Implementation for Linear Elasto-statics

4.1 Interpolation by CLSA

The interpolations for the displacements u_k, $k = 1,2,3$, of the direction x, y and z are given in by CLSA as (43).

$$u_k = \sum_{i,j} d_{ij}^k N_{ij}^k, \quad k = 1,2,3 \quad (43)$$

in which, k is the index for direction x, y and z; i is the index of covers; and j is the index of unknowns of each cover function. The superscript means the direction x, y and z by 1, 2 and 3.

The displacement vector can be expressed as

$$\mathbf{u} = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = \begin{bmatrix} \mathbf{N}_1 \mid \mathbf{N}_2 \mid \cdots \mid \mathbf{N}_{NC} \end{bmatrix} \begin{Bmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \vdots \\ \mathbf{d}_{NC} \end{Bmatrix} \equiv \mathbf{Nd} \quad (44)$$

where

$$\mathbf{N}_i = \begin{bmatrix} N_{i0} & N_{i1} & \cdots & N_{i,n_i} & 0 & & & 0 \\ 0 & & & 0 & N_{i0} & N_{i1} & \cdots & N_{i,n_i} & 0 \\ 0 & & & 0 & & 0 & & N_{i0} & N_{i1} & \cdots & N_{i,n_i} \end{bmatrix} \quad i = 1,2,\cdots NC$$

$$\mathbf{d}_i = \{ d_{i0}^1 \quad d_{i1}^1 \quad \cdots \quad d_{i,n_i}^1 \mid d_{i0}^2 \quad d_{i1}^2 \quad \cdots \quad d_{i,n_i}^2 \mid d_{i0}^3 \quad d_{i1}^3 \quad \cdots \quad d_{i,n_i}^3 \}^T \quad i = 1,2,\cdots NC$$

Then assuming that the initial stress and strain are 0,

the strain and stress can be written as

$$\begin{aligned}\varepsilon &= \mathbf{Bd} \\ &= [\mathbf{B}_1 \mid \mathbf{B}_2 \mid \cdots \mid \mathbf{B}_{NC}]\mathbf{d}\end{aligned} \tag{45}$$

$$\sigma = \mathbf{D}\varepsilon = \mathbf{DBd} \tag{46}$$

where

$\mathbf{D}$ is the stress-strain matrix, and

$$\mathbf{B}_i = \begin{bmatrix} \frac{\partial N_{i0}}{\partial x} & \frac{\partial N_{i1}}{\partial x} & \cdots & \frac{\partial N_{i,n_i}}{\partial x} & 0 & & 0 & & \\ 0 & & & \frac{\partial N_{i0}}{\partial y} & \frac{\partial N_{i1}}{\partial y} & \cdots & \frac{\partial N_{i,n_i}}{\partial y} & 0 & \\ 0 & & & 0 & & & \frac{\partial N_{i0}}{\partial z} & \frac{\partial N_{i1}}{\partial z} & \cdots & \frac{\partial N_{i,n_i}}{\partial z} \\ \frac{\partial N_{i0}}{\partial y} & \frac{\partial N_{i1}}{\partial y} & \cdots & \frac{\partial N_{i,n_i}}{\partial y} & \frac{\partial N_{i0}}{\partial x} & \frac{\partial N_{i1}}{\partial x} & \cdots & \frac{\partial N_{i,n_i}}{\partial x} & 0 \\ 0 & & & \frac{\partial N_{i0}}{\partial z} & \frac{\partial N_{i1}}{\partial z} & \cdots & \frac{\partial N_{i,n_i}}{\partial z} & \frac{\partial N_{i0}}{\partial y} & \frac{\partial N_{i1}}{\partial y} & \cdots & \frac{\partial N_{i,n_i}}{\partial y} \\ \frac{\partial N_{i0}}{\partial z} & \frac{\partial N_{i1}}{\partial z} & \cdots & \frac{\partial N_{i,n_i}}{\partial z} & 0 & & \frac{\partial N_{i0}}{\partial x} & \frac{\partial N_{i1}}{\partial x} & \cdots & \frac{\partial N_{i,n_i}}{\partial x} \end{bmatrix} \tag{47}$$

4.2 Principle of virtual work

Consider a general elasto-statics problem with the body force $\bar{\mathbf{b}}$ acting in the domain V with the boundary $S = S_1 + S_2$. On the boundary S_1 the traction is equal to $\bar{\mathbf{t}}$, and the displacement on the boundary S_2 is equal to $\bar{\mathbf{u}}$.

$$\sigma_{ij,j} = \bar{b}_i \quad \text{in } V + S \tag{48}$$

$$\mathbf{t} = \bar{\mathbf{t}} \quad \text{on } S_1 \tag{49}$$

$$\mathbf{u} = \bar{\mathbf{u}} \quad \text{on } S_2 \tag{50}$$

By the principle of virtual work, the elasto-statics problem can be described as the following equation, in which the virtual displacement $\delta\mathbf{u}$ is arbitrary satisfying the condition (50) and $\delta\varepsilon$ is the virtual strain caused by $\delta\mathbf{u}$.

$$\int_V \{\delta\varepsilon\}^T \sigma dV = \int_{S_1} \delta\mathbf{u}\bar{\mathbf{t}} dS + \int_V \delta\mathbf{u}\bar{\mathbf{b}} dV \tag{51}$$

Substituting (44), (45) and (46) into (51) and considering that the virtual displacement is arbitrary, we have

$$\int_V \mathbf{B}^T\mathbf{DB}dV * \mathbf{d} = \int_{S_1} \mathbf{N}\bar{\mathbf{t}}dS + \int_V \mathbf{N}\bar{\mathbf{b}}dV \tag{52}$$

As same as FEM, we define

$$\mathbf{K} = \int_V \mathbf{B}^T\mathbf{DB}dV \tag{53}$$

$$\mathbf{F} = \int_{S_1} \mathbf{N}\bar{\mathbf{t}}dS + \int_V \mathbf{N}\bar{\mathbf{b}}dV \tag{54}$$

Then the stiffness equation can also be written in the same format as FEM, but $\mathbf{d}$ does not mean the node displacement vector.

$$\mathbf{Kd} = \mathbf{F} \tag{55}$$

4.3 Element stiffness matrix

Since the voxel data based elements are available, the integrals of (53) can be evaluated simply on each element. Moreover, because the support of every cover is defined as element-wise, the shape functions are also limited as element-wise. That is, on each element only the covers distributed on its sides are involved in the evaluation of the element stiffness matrix. This implies that the global stiffness matrix is banded just like FEM.

$$\mathbf{K} = \int_V \mathbf{B}^T\mathbf{DB}dV = \sum_{l=1}^{NE} \int_{V_l^{(e)}} \mathbf{B}_l^{(e)T}\mathbf{DB}_l^{(e)}dV = \sum_{l=1}^{NE} \mathbf{K}_l^{(e)} \tag{56}$$

where NE is the total number of elements. $V_l^{(e)}$ is the physical domain of the l th element.

$$\mathbf{K}_l^{(e)} = \int_{V_l^{(e)}} \mathbf{B}_l^{(e)T}\mathbf{DB}_l^{(e)}dV \tag{57}$$

4.4 Handling of loads

The load vector can also be evaluated at the level of element. Since the shape functions are limited as element-wise as discussed above, when integrating on the load boundary within one element only the covers allocated on the sides of the element are involved. It is the same for the body force.

$$\mathbf{F} = \int_{S_1} \mathbf{N}\bar{\mathbf{t}}dS + \int_V \mathbf{N}\bar{\mathbf{b}}dV = \sum_{l=1}^{NE} \left(\int_{S_{1l}^{(e)}} \mathbf{N}_l^{(e)}\bar{\mathbf{t}}dS + \int_{V_l^{(e)}} \mathbf{N}_l^{(e)}\bar{\mathbf{b}}dV \right) \tag{58}$$

where $S_{1l}^{(e)}$ is the part of load boundary included in the l th element.

4.5 Handling of displacement conditions

In the current implementation, the displacement

conditions are dealt with by the penalty method having the physical interpretation of the stiff spring constraint. Integrating the product of (50) and the penalty factor k on the displacement boundary S_2 and adding to (55), we have

$$(\mathbf{K} + \int_{S_2} k\mathbf{N}dS)\mathbf{d} = \mathbf{F} + \int_{S_2} k\bar{\mathbf{u}}dS \qquad (59)$$

It can also be handled at the level of element as same as the load conditions.

$$(\mathbf{K} + \sum_{l=1}^{NE} \int_{S_{2l}^{(e)}} k\mathbf{N}_l^{(e)}dS)\mathbf{d} = \mathbf{F} + \sum_{l=1}^{NE} \int_{S_{2l}^{(e)}} k\bar{\mathbf{u}}dS \qquad (60)$$

where $S_{2l}^{(e)}$ is the part of displacement boundary included in the l th element.

All the integrals are carried out on physical covers. On the elements not fully inside the physical domain, the integration is calculated only on the inside part. The implement detail can be found in Reference 17.

5. Numerical examples

5.1 Constant stress cube

The following conditions are imposed on a cube as shown in Figure 3 ($E = 1, \nu = 0.3$).

Displacement conditions

$$u_x = 0 \quad \text{on} \quad x = 0$$
$$u_y = 0 \quad \text{on} \quad y = 0$$
$$u_z = 0 \quad \text{on} \quad z = 0$$

Traction condition

$$P_z = 1 \quad \text{on} \quad z = 1$$

The covers are distributed based on a two-level voxel data as shown in Figure 4.

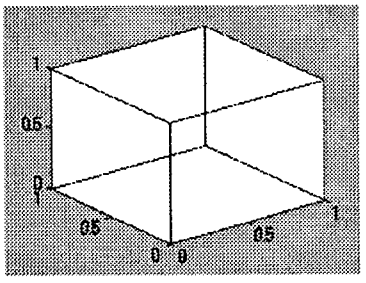


Figure 3 A cube

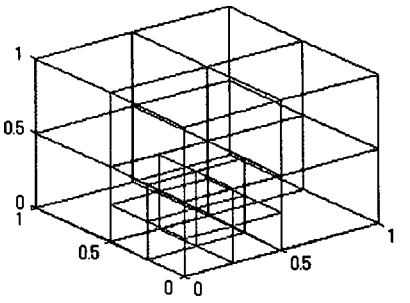


Figure 4 Two-level resolution voxel data based model

As shown in Figure 5 the computed results of displacement is linear and the stress is constant everywhere, and it is proven that the CLSA can give an exact solution for constant stress problem

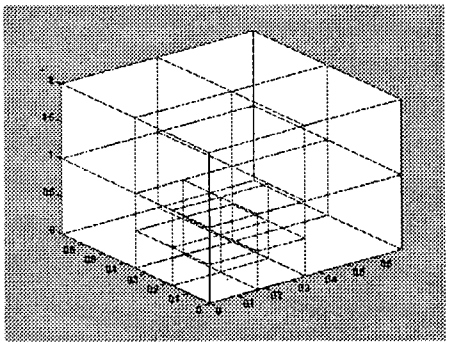


Figure 5 Displacement result for two-level resolution voxel data based model

5.2 Plate with a hole

Consider a 20×20×4 plate with a hole at its center, whose radius is 2. Uniform traction acts on a pair of opposite sides. Figure 6 shows 1/8 of it. The boundary conditions are given as

$$u_x = 0 \quad \text{on} \quad x = 0$$
$$u_y = 0 \quad \text{on} \quad y = 0$$
$$u_z = 0 \quad \text{on} \quad z = 0$$
$$\sigma_{yy} = 1 \quad \text{on} \quad y = 10$$

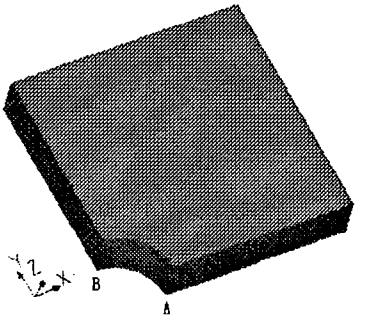
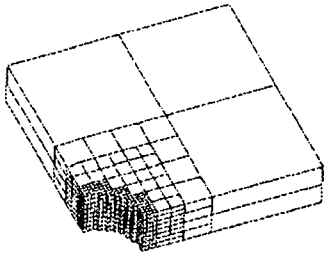
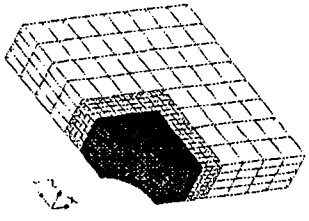


Figure 6 1/8 plate with a hole

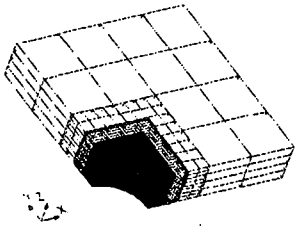
Three Models as shown in Figure 7 are calculated. In the Model a, four-level voxel data is employed to create the cover distribution. The sizes of four level voxels are respectively 10/2×10/2×2/2, 10/4×10/4×2/4, 10/16×10/16×2/8 and 10/64×10/64×2/8. In the Model b, three-level voxel data is employed to create the cover distribution. The sizes of the three level voxels are respectively 10/8×10/8×2/4, 10/32×10/32×2/8, and 10/128×10/128×2/16. In the Model c, four-level voxel data is employed to create the cover distribution. The sizes of the four level voxels are respectively 10/4×10/4×2/4, 10/16×10/16×2/8, 10/64×10/64×2/16, and 10/256×10/256×2/32.



(a)

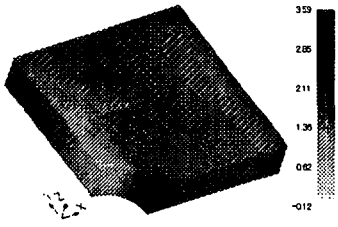


(b)

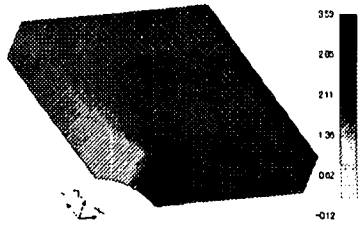


(c)

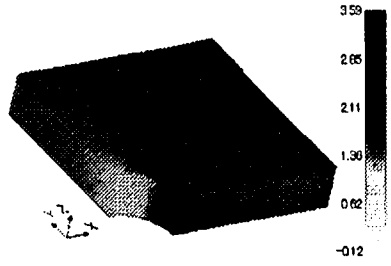
Figure 7 Three models for the plate with a hole



(a)



(b)



(c)

Figure 8 Computed results of σ_{yy}

The computed results of σ_{yy} are shown in Figure 8. The stress values at point A and B (see Figure 6) are listed in Table 1 together with the ANSYS results and the numbers of DOF are listed in Table 2.

Table 1 Stress values

		σ_{xx}	σ_{yy}	σ_{zz}	Von Mises stress
Point A	Model a	----	3.269	0.382	3.092
	Model b	----	3.339	0.405	3.116
	Model c	----	3.464	0.420	3.189
	ANSYS	----	3.578	0.398	3.389
Point B	Model a	-1.287	----	-0.387	----
	Model b	-1.458	----	-0.416	----
	Model c	-1.502	----	-0.428	----
	ANSYS	-1.488	----	-0.405	----

Table 2 DOF

Model	a	b	c	ANSYS
DOF	4257	101337	399966	12555

The above results show that a good local approximation can be achieved by the CLSA with multi-resolution voxel data based cover distribution, instead of increasing the number of degrees of

freedom globally. It is demonstrated that the analysis accuracy can be controlled by locally adjusting the density of cover distribution.

6. Conclusion

In this paper, the Cover Least Square Approximation method is implemented for linear structure analysis. The covers are distributed using multi-resolution voxel data. Numeric examples show that multi-resolution voxel data based cover distribution can conveniently guarantee the linear independence of CLSA shape functions and the approximation accuracy can be flexibly controlled by locally justifying the density of cover distribution.

Since CLSA itself is in its infancy, it still requires considerable improvement. For example, the a posteriori error estimation and adaptive refining process are necessary for fully automatic analysis and only after the adaptive refining process is implemented the efficiency of the presented method can appear.

REFERENCES

(1) T. Liszka and J. Orkisz, The finite difference method at arbitrary irregular grids and its applications in applied mechanics, Comput. Struct., 11 (1980) 83-95.

(2) J.J. Monaghan, An introduction to SPH, Comput. Phys. Commun., 48 (1982) 89-96.

(3) B. Nayroles, G. Touzot and P. Villon, Generiazing the finite elment method: diffuse approximation and diffuse elements, Comput. Mech., 10 (1992) 307-318.

(4) E.J. Kansa, Multiquadrics – A scattered data approximation scheme with application to computational fluid dynamics: I . Surface approximations and partial derivative estimates, Comput. Math. Applic., 19 (1990) 127-145

(5) E.J. Kansa, Multiquadrics – A scattered data approximation scheme with application to computational fluid dynamics: II . Solution to parabolic, hyperbolic and elliptic partial differential equarions, Comput. Math. Applic., 19 (1990) 147-161

(6) Belytschko, T., Lu, Y.Y and Gu, L., Element free Galerkin method, Int. J. Numer. Methods Engrg., Vol.37, pp229-256, 1994

(7) S. Qian and J. Weiss, Wavelet and the numerical solution of partial differential equations, J. Comput. Phys., 106 (1993) 155-175.

(8) C.A. Duarte and J.T. Oden, Hp cloud – A meshless method to solve boundary-value problems, Technical Report 95-05, Texas Institute for Computational and Applied Mathematics, Austin, 1995.

(9) W.K. Liu, Sukky Jun, S. Li, J. Adee and T. Belytschko, Reproducing kernel particle methods for structural dynamics, Int. J. Numer. Methods Engrg., 38 (1995) 1655-1679.

(10) Babuska, I. and Menlenk, J.M., The partition of unity finite element method, Int. J. Numer. Methods Engrg., Vol.40, pp727-758, 1997

(11) T. Belytschko, Y. Krongauz, D. Organ, M. Fleming and P. Krysl, Meshless method: An overview and recent developments, Comput. Methods Appl. Mech. Engrg., 139 (1996),3-47

(12) C.R. Jin, K. Nakanishi, K. Suzuki and H. Ohtsubo, On the independence of approximation function in finite cover method, Proceeding of the Conference on Computational Engineering and Science, Vol.3, No.2, pp381-384 (1998)

(13) P. Lancaster and K. Salkauskas, Surface generated by moving least squares methods, Mathematics of Computer, Vol. 37, No. 155, 1981.

(14) C.R. JIN, K. Susuki, D. Fujii and H. Ohtsubo, Methodology and Property of Cover Least Square Approximation, Transactions of the Japan Society for Computational Engineering and Science, Vol.5, 2000.

(15) Shi, G. H., Manifold method of material analysis, Transactions of the 9th Army Conference On Applied Mathematics and Computing, Report No. 92-1. U.S. Army Research Office, 1991

(16) C.R. Jin, Multi-scale Cover Method Using Voxel Information, Doctor Thesis of Tokyo University, 2000.

(17) K. Suzuki, H. Ohtsubo and C. Jin, The Analysis of 3D Solid Using Multi-scale Voxel Data, Computational Mechanics -New Trends and Applications Part Ⅶ, Section 2-15, E. Onate and S.R. Idelsohn (Eds.) CIMNE, Barcelona, Spain 1998.

(Received April 21, 2000)