Computational Mechanics 31 (2003) 27–37 © Springer-Verlag 2003
DOI 10.1007/s00466-002-0390-3

On the method of finite spheres in applications: towards the use with ADINA and in a surgical simulator

S. De, J.-W. Hong, K. J. Bathe

Abstract In this paper we report some recent advances regarding applications using the method of finite spheres; a truly meshfree numerical technique developed for the solution of boundary value problems on geometrically complex domains. First, we present the development of a preprocessor for the generation of nodal points on two-dimensional computational domains. Then, the development of a specialized version of the method of finite spheres using point collocation and moving least squares approximation functions and singular weight functions is reported for rapid computations in virtual environments involving multi-sensory (visual and touch) interactions.

Keywords Method of finite spheres, Meshfree method, ADINA, Surgical simulation

1 Introduction

In this paper we present some recent developments in the method of finite spheres [1]; a truly meshfree numerical technique developed for the solution of boundary value problems on complex domains. Meshfree methods are attractive since they circumvent the problem of mesh generation and offer greater flexibility in the choice of approximation spaces and the use of weighted residual schemes than the finite element/finite volume techniques. Over the past several years quite a few meshfree techniques have been proposed as was reviewed in ref. [1].

However, for a meshfree method to be competitive with the finite element/finite volume techniques, it must be computationally efficient. The method of finite spheres has been developed with the considerations of computational efficiency and reliability in mind [1, 2]. In this technique, the idea of a finite element discretization is generalized to a sphere. The computational domain is discretized using a set of nodal points and the interpolation functions are compactly supported on spheres surrounding these nodes. A weighted residual scheme [3] is employed as the error minimization criterion.

In [4] effective schemes for incorporating the Dirichlet boundary conditions in the weak sense were introduced and in [1] efficient techniques of performing numerical integration of the terms in the Galerkin weak form were presented. However, results were reported for problems having simple two-dimensional geometries. In this paper we present techniques of automatically generating the open cover for two-dimensional domains having complex geometries by generating the nodal points and computing the sphere radii.

In most engineering applications we require numerical techniques that are robust and reliable and computationally efficient. But real time performance is seldom an issue. For certain applications, however, real time performance is critical. One such application is the development of medical simulators to train physicians. In addition to the computer such systems have one or more human machine interface(s). In order to be realistic, they should provide *multi sensory* interaction capabilities and perform in real time.

An example of medical procedures where both visual and tactile (force) interactions are important is minimally invasive surgery. Laparoscopic surgery is a particular example of such a procedure. This technology uses a small video camera and a few long slender instruments to perform surgery with minimum incision. The camera and instruments are introduced into the abdomen or chest through small skin incisions that enable the surgeon to explore and operate on the internal cavity without the need for making large openings. Laparoscopic surgery is a revolutionary technique but requires intensive training. The success of flight simulators in training pilots has inspired the development of analogous "surgical simulators" as immersive virtual environment systems that will train medical personnel with virtual patients [5].

For real time visual display an update rate of 30 Hz is sufficient. For haptic display, the Phantom (developed by Sensable Technologies, Inc.) haptic interface device is used. This is a low inertia device that has six displacement degrees of freedom to capture the orientation and position of the users hands. The paradigm of interaction with the virtual environment is similar to the exploration of the world using a walking stick. When the virtual Phantom tool tip interacts with a virtual object, the interface can

S. De (✉)
Department of Mechanical,
Aerospace and Nuclear Engineering,
Rensselaer Polytechnic Institute,
110 8th Street, Troy, NY 12180

J.-W. Hong, K. J. Bathe
Department of Mechanical Engineering,
Massachusetts Institute of Technology,
77 Massachusetts Avenue, Cambridge, MA 02139

Dedicated to the memory of Prof. Mike Crisfield, for his cheerfulness and cooperation as a colleague and friend over many years.

deliver the three components of the interaction force to the user resulting in the sensation of touch. For stable simulation, the haptic loop requires an update rate of about 1 kHz [6, 7]. This imposes severe restrictions on the complexity of the models that can be rendered.

Various techniques can be found in literature for the simulation and display of deformable objects. These techniques can be categorized into two main approaches: "geometrically based" approaches and "physically based" approaches (see [8] for a review). The "geometrically based" modeling approaches, such as Bezier/B-spline based procedures and free form deformation techniques, do not account for the physics of deformation, but are simpler to implement. In contrast, the "physically based" approaches, such as the lumped parameter particle models and finite element based techniques, attempt to model the underlying physics, but are computationally intensive.

One of the most popular "physically based" modeling approaches is a lumped parameter technique using masses, springs and dampers. Each node has a point mass associated with it and is connected to its neighbors using linear or nonlinear springs and dampers. Cover et al. developed the first laparoscopic gall bladder surgery simulator using surface-based spring-mass models [9]. The spring-mass models are also widely used in facial simulations. For example, Terzopoulos and Waters constructed a three-layer network of springs based on three anatomically distinct layers of facial tissue [10].

The spring-mass models are simple and computationally very efficient. However, for purpose of soft-tissue deformation modeling, they suffer from the following problems: (1) it is difficult to design a topologically admissible network of springs and masses so as to prevent the system to fall into unwanted local minima; (2) it is difficult, and sometimes impossible, to determine the parameters of a large number of springs, dampers and masses to represent the global behavior of the tissue especially if it is intended to capture nonlinear and/or viscoelastic behavior; and finally (3) it is difficult to enforce global properties like incompressibility when using such models.

To overcome the problems associated with the spring-mass models, Pieper et al. developed a planning system for facial plastic surgery using isoparametric finite elements [11]. Bro-Nielsen et al. applied finite elements for real time surgical simulations using tetrahedral volume elements [12]. The computation time was reduced significantly by using a matrix condensation technique. Cotin et al. demonstrated a hepatic surgery simulator using similar finite element models [13, 14].

However, finite element techniques suffer from certain drawbacks in real time simulations: (1) the need for numerical integration and volumetric meshing results in a slower-than-real-time performance unless extensive precomputations are performed; (2) the contact between tool and tissues must occur only at nodal points, hence, to prevent loss of resolution, the density of nodal points should be sufficiently high thus requiring extensive memory resources and high computational overhead; and finally, (3) cutting or tearing requires an expensive remeshing process during simulation, hence precomputed data of the object becomes, at least locally, invalid and all the data displayed to the user must be computed in real time.

In this paper we discuss the development of a fast version of the method of finite spheres for the purpose of real time simulations of surgical procedures in virtual environments. The approximation functions are developed using singular weight functions and the moving least squares technique [2]. A point collocation scheme is used and therefore no numerical integration is necessary. This technique is different from the finite point method [15], which uses a weighted least squares technique for generating the approximation functions.

In Sect. 2 we recapitulate both the displacement and displacement/pressure mixed formulations of the Galerkin-based method of finite spheres and present techniques of automatically generating an open cover for two-dimensional geometries. In Sect. 3 we discuss the specialized version of the method of finite spheres using point collocation. In Sect. 4 we present several examples demonstrating the applicability of the proposed techniques.

2
The Galerkin-based method of finite spheres

2.1
Approximation functions

In this section we briefly recapitulate how we generate low-cost approximation functions using the partition of unity paradigm [16] based on the Shepard partition of unity functions [17].

Let $\Omega \in R^d$ ($d = 1, 2$ or 3) be an open bounded domain and let Γ be its boundary (see Fig. 1). Let a family of open spheres $\{B(\mathbf{x}_I, r_I); I = 1, 2, \ldots, N\}$ form a covering for Ω, i.e., $\Omega \subset \bigcup_{I=1}^{N} B(\mathbf{x}_I, r_I)$, where $\mathbf{x}_I$ and r_I refer to the center and radius of the sphere I, respectively. We associate a "node" with the geometric center of each sphere. By $S(\mathbf{x}_I, r_I)$ we denote the surface of sphere I. The spheres may be entirely within the domain (interior spheres) or may have nonzero intercepts with the boundary (boundary spheres), see Fig. 1.

We define a radial weighting function $W_I(\mathbf{x})$, of the form $W_I(\mathbf{x}) = W(s_I)$, where $s_I = \| \mathbf{x} - \mathbf{x}_I \|_0 / r_I$, compactly supported on the sphere centered at node I. We have chosen a quartic spline weighting function of the following form:

$$W(s_I) = 1 - 6s_I^2 + 8s_I^3 - 3s_I^4; \quad 0 \le s_I \le 1 \qquad (1)$$

The weighting functions define the Shepard partition of unity functions [17]

$$\varphi_I^0(\mathbf{x}) = \frac{W_I}{\sum_{J=1}^{N} W_J}, \quad I = 1, 2, \ldots, N$$

The functions $\{\varphi_I^0(\mathbf{x})\}$ satisfy zeroth order consistency. To generate approximation spaces with higher order consistency, a local approximation space $V_I^h = \text{span}_{m \in \mathcal{I}}\{p_m(\mathbf{x})\}$ is defined at each node I, where $p_m(\mathbf{x})$ is a polynomial or other function and $\mathcal{I}$ is an index set. The superscript h is a measure of the size of the spheres.

The global approximation space V_h is generated by multiplying the partition of unity function at each node I with the functions from the local basis

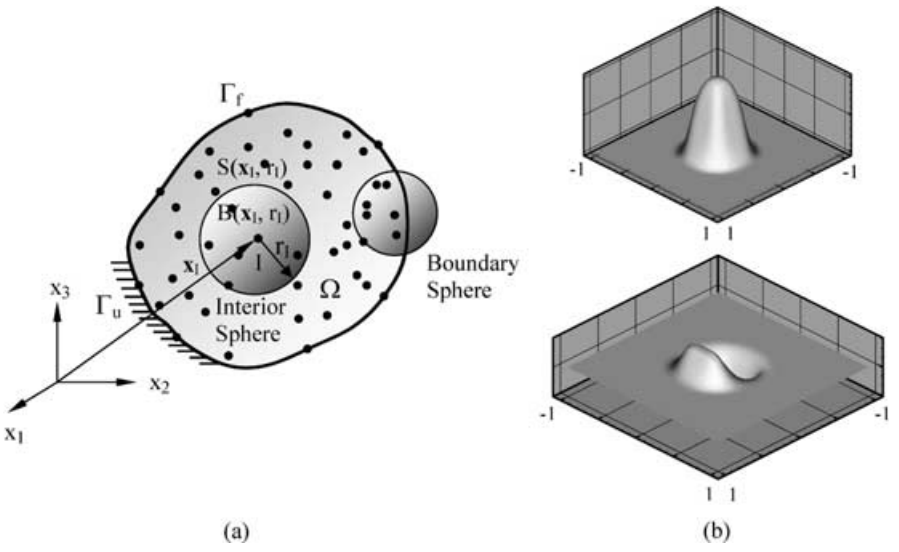


Fig. 1. a A schematic of the method of finite spheres and **b** some shape functions in two-dimensions

$$V_h = \sum_{I=1}^{N} \varphi_I^0 V_I^h$$

Hence, any function $v_h \in V_h$ can now be written as

$$v_h(\mathbf{x}) = \sum_{I=1}^{N} \sum_{m \in \mathcal{I}} h_{Im}(\mathbf{x}) \alpha_{Im} \qquad (2)$$

where

$$h_{Im}(\mathbf{x}) = \varphi_I^0(\mathbf{x}) p_m(\mathbf{x}) \qquad (3)$$

is a basis/shape function associated with the mth degree of freedom α_{Im} of node I.

2.2 Automatic generation of nodes on an arbitrary domain in R^2

An important practical issue is the generation of the approximation spaces on geometries of complex shapes. Unlike the functions generated using the moving least squares technique (see [2]) the only criterion that needs to be satisfied for the functions used in the method of finite spheres is that the computational domain is a subset of the union of the spheres. Of course, an additional problem is to compute the intersections of the spheres with the domain boundary. An arbitrary distribution of nodal points on the domain would render these tasks difficult and inefficient and therefore some structure is essential. In [18] one such structure is defined using an octree based approach for the domain and a tetrahedral mesh at the boundary. An extension of this approach to the method of finite spheres is presented in [19] which uses a quadtree/octree representation of the domain with special techniques being applied to the boundary spheres.

In the present paper, however, we adopt an alternative approach and present a technique of generating the nodal points and defining the interior and boundary spheres using a commercial finite element analysis program, ADINA (ADINA R&D, Inc., http://www.adina.com). It is, therefore, possible to take advantage of the powerful node generating algorithms of ADINA and compare the results

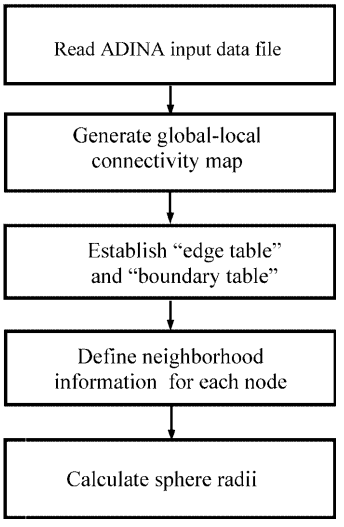


Fig. 2. Flow chart for generating a cover for a two-dimensional domain

obtained with the method of finite spheres directly with that provided by ADINA for exactly the same nodal configurations. It should be noted, however, that the finite element program is only used to provide a structure to the distribution of nodes on the domain and the underlying simplices are not used for interpolation or integration. The structure provided by the finite element program provides nearest neighbor information in constant time. Furthermore, unlike in "pseudo meshfree" techniques, the integration points in the method of finite spheres are uniquely associated with each sphere or the region of intersection of two spheres, hence it is not necessary to search for the spheres contributing to a given integration point.

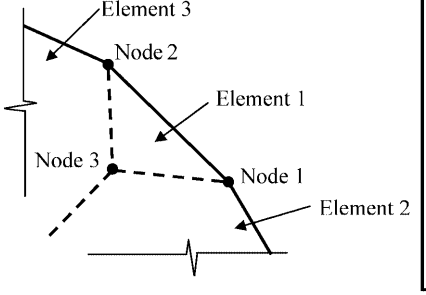
Figure 2 shows a flowchart of how the ADINA input file may be used to develop an open cover for the method of finite spheres. The program STAP [3] has been modified to MFS_STAP to accept the nodal information generated by ADINA. The first step involves the creation of an "edge table" and a "boundary table". The information stored in

these two tables is used in determining the radii of the spheres as well as the intersection of the spheres with domain boundaries.

In the ADINA data file each element has three or four edges with each edge having two nodes. The process of generation of the "edge table" from the finite element program is explained in Fig. 3. This table has the structure (N1, N2, flipping flag), where N1 and N2 are the nodes along a finite element edge (with N1 < N2). If the local connectivity information of the nodes necessitates N1 > N2 then the "flipping flag" is set to unity otherwise it is zero. Note that for an edge interior to the domain (in R^2) the information in the edge table is presented twice with the value of the "flipping flag" different in the two cases. Edge table information is not repeated for edges representing the domain boundary. The information regarding the boundary edges is stored in the "boundary table". The edge table also provides, for each node, information regarding the nearest neighbors since these are the nodes connected to that node along the edges emanating from it.

The next step is to compute the radii of the spheres. Figure 4 shows the construction of a boundary sphere. The radius of the sphere at a node lying on the boundary is obtained as the smaller of the edge lengths connecting this node to its two nearest neighbors. It is to be noted that the node generation scheme of ADINA does not allow a large difference between adjacent edge lengths and therefore this scheme ensures a satisfactory cover. The boundary table may be utilized to compute the included angle between the two edges and this information is used for the generation of integration points for the boundary sphere (see Fig. 4).

For a node which, together with its nearest neighbors, is within the domain, the radius of the sphere is computed as the average of the nearest neighbor distances as obtained from the edge table. For a node which is within the domain, but has at least two of its nearest neighbors on the boundary, the minimum normal distance from the nearest boundary edges is adopted as the sphere radius (see Fig. 5).



N1	N2	Flag
1	2	0
2	3	0
1	3	1
		•
		•
1	3	0
2	3	1
		•
		•

Fig. 3. Edge table generation scheme. Each row of the edge table contains information about the nodes that are on an element edge. The format is (N1, N2, "flipping flag"), where N1 and N2 are the nodes connected by the edge and the flipping flag takes on a value of zero if the order is N2 < N1 in the original connectivity information of the element and a value of 1 is the original order is reversed

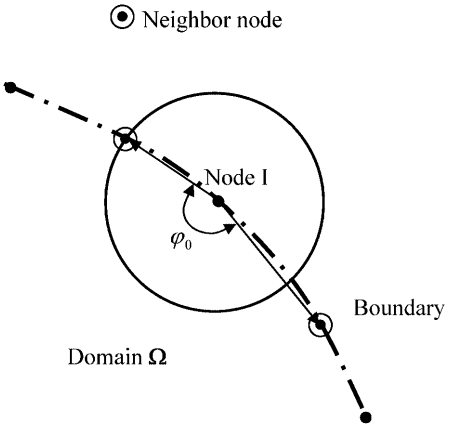


Fig. 4. Generating a boundary sphere at node 1. The boundary nodes are obtained from the boundary table. The radius of the sphere at a boundary node is selected as the minimum of the two nearest neighbor distances. The included angle between the two edges (φ_0) is used in the numerical integration scheme for this sphere

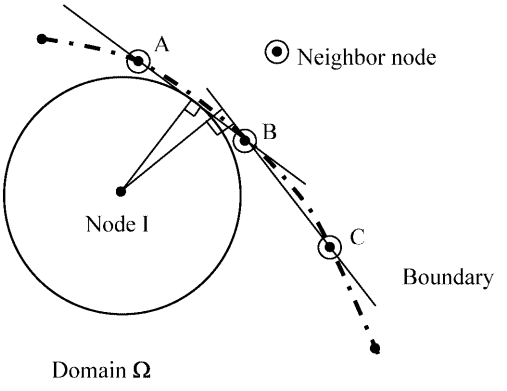


Fig. 5. A node (I) that has at least two neighbors that are on the domain boundary needs special attention. The normal distances of such a node from the neighboring edges (AB and BC in the above example) are computed and the radius of the sphere at node I is assumed as the minimum of these distances

2.3 Displacement-based formulation

We consider the following variational problem from linear elasticity.

Find $\mathbf{u} \in H^1(\Omega)$ such that

$$\int_\Omega \boldsymbol{\epsilon}^T(\mathbf{v})\mathbf{C}\boldsymbol{\epsilon}(\mathbf{u})\,\mathrm{d}\Omega - \int_{\Gamma_u} \left[\boldsymbol{\epsilon}^T(\mathbf{v})\mathbf{C}\mathbf{N}^T\mathbf{u} + \mathbf{v}^T\mathbf{N}\mathbf{C}\boldsymbol{\epsilon}(\mathbf{u})\right]\mathrm{d}\Gamma = \int_\Omega \mathbf{v}^T\mathbf{f}^B\,\mathrm{d}\Omega + \int_{\Gamma_f} \mathbf{v}^T\mathbf{f}^S\,\mathrm{d}\Gamma - \int_{\Gamma_u} \boldsymbol{\epsilon}(\mathbf{v})^T\mathbf{C}\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \quad \forall\ \mathbf{v} \in H^1(\Omega) \tag{4}$$

where $H^1(\Omega)$ is the first order Hilbert space [3]. $\mathbf{u}$ and $\boldsymbol{\epsilon}$ are the displacement and strain vectors, $\mathbf{C}$ is the elasticity matrix, $\mathbf{f}^S$ is the prescribed traction vector on the

Neumann boundary Γ_f, $\mathbf{u}^S$ is the vector of prescribed displacements on the Dirichlet boundary Γ_u (note that the domain boundary $\Gamma = \Gamma_f \cup \Gamma_u$), $\mathbf{f}^B$ is the body force vector (including inertia terms), and $\mathbf{N}$ is the matrix of direction cosine components of a unit normal to the domain boundary (positive outwards).

We have the following approximation for the displacement field

$$\mathbf{u}(x,y) = \sum_{J=1}^{N} \sum_{n\in\mathcal{I}} \mathbf{H}_{Jn}(x,y)\boldsymbol{\alpha}_{Jn} = \mathbf{H}(x,y)\mathbf{U} \tag{5}$$

where

$$\mathbf{U} = [\boldsymbol{\alpha}_{10}\ \boldsymbol{\alpha}_{11}\ \boldsymbol{\alpha}_{12}\cdots\boldsymbol{\alpha}_{Jn}\cdots]^{\mathrm{T}}$$

is the vector of nodal unknowns, and

$$\boldsymbol{\alpha}_{Jn} = [u^{Jn} \quad v^{Jn}]$$

is the vector of nodal unknowns at node J corresponding to the nth degree of freedom (u^{Jn} and v^{Jn} are the nodal variables for the x and y direction displacements at node J corresponding to the nth degree of freedom). The nodal shape function matrix corresponding to the nth degree of freedom is

$$\mathbf{H}_{Jn}(x,y) = \begin{bmatrix} h_{Jn}(x,y) & 0 \\ 0 & h_{Jn}(x,y) \end{bmatrix} \tag{6}$$

We obtain the discretized system of algebraic equations corresponding to node I and degree of freedom m

$$\sum_{J=1}^{N} \sum_{n\in\mathcal{I}} \mathbf{K}_{ImJn}\boldsymbol{\alpha}_{Jn} = \mathbf{f}_{Im} + \hat{\mathbf{f}}_{Im} \tag{7}$$

In this equation the various matrices and vectors are as follows:

$$\mathbf{K}_{ImJn} = \int_{\Omega_I} \mathbf{B}_{Im}^T \mathbf{C}\mathbf{B}_{Jn}\,\mathrm{d}\Omega \tag{8}$$

$$\mathbf{f}_{Im} = \int_{\Omega_I} \mathbf{H}_{Im}\mathbf{f}^B\,\mathrm{d}\Omega \tag{9}$$

where $\Omega_I = \Omega \cap B(\mathbf{x}_I, r_I)$ and $\mathbf{B}(x,y)$ is the strain–displacement matrix.

If I is a node associated with an "internal sphere", then, due to compact support

$$\hat{\mathbf{f}}_{Im} = \mathbf{0}$$

If the sphere corresponding to node I has nonzero intercept on the Neumann boundary, then

$$\hat{\mathbf{f}}_{Im} = \int_{\Gamma_{f_I}} \mathbf{H}_{Im}\mathbf{f}^S\,\mathrm{d}\Gamma \tag{10}$$

where $\Gamma_f = \cup_{I\epsilon N_f}\ \Gamma_{fI}$, $\mathcal{N}_f$ being the index set of such nodes.

On the other hand, if the sphere corresponding to node I has nonzero intercept on the Dirichlet boundary, then

$$\hat{\mathbf{f}}_{Im} = \sum_{J=1}^{N} \sum_{n\in\mathcal{I}} \tilde{\mathbf{K}}_{ImJn}\boldsymbol{\alpha}_{Jn} - \tilde{\mathbf{f}}_{Im} \tag{11}$$

where

$$\tilde{\mathbf{K}}_{ImJn} = \int_{\Gamma_{u_I}} \mathbf{H}_{Im}\mathbf{N}\mathbf{C}\mathbf{B}_{Jn}\,\mathrm{d}\Gamma + \int_{\Gamma_{u_I}} \mathbf{B}_{Im}^T\mathbf{C}\mathbf{N}^T\mathbf{H}_{Jn}\,\mathrm{d}\Gamma \tag{12}$$

and

$$\tilde{\mathbf{f}}_{Im} = \int_{\Gamma_{u_I}} \mathbf{B}_{Im}^T\mathbf{C}\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \tag{13}$$

where $\Gamma_u = \cup_{I\in\mathcal{N}_u}\Gamma_{u_I}$, $\mathcal{N}_u$ being the index set of such nodes.

2.4 Displacement/pressure mixed formulation

For an almost incompressible medium in plane strain conditions, we consider the following weak form

Find $\mathbf{u} \in H^1(\Omega)$ and $p \in L^2(\Omega)$ such that

$$\begin{aligned}
&\int_{\Omega} \left[\boldsymbol{\epsilon}^{D^T}(\mathbf{v})\mathbf{C}^D\boldsymbol{\epsilon}^D(\mathbf{u}) - \epsilon_V(\mathbf{v})p\right]\mathrm{d}\Omega \\
&\quad - \int_{\Gamma_u} \left[\boldsymbol{\epsilon}^{D^T}(\mathbf{v})\mathbf{C}^D\mathbf{N}^T\mathbf{u} + \mathbf{v}^T\mathbf{N}\mathbf{C}^D\boldsymbol{\epsilon}^D(\mathbf{u})\right]\mathrm{d}\Gamma + \int_{\Gamma_u} \mathbf{v}^T\mathbf{N}\mathbf{I}p\,\mathrm{d}\Gamma \\
&\quad = \int_{\Omega} \mathbf{v}^T\mathbf{f}^B\,\mathrm{d}\Omega + \int_{\Gamma_f} \mathbf{v}^T\mathbf{f}^S\,\mathrm{d}\Gamma \\
&\quad\quad - \int_{\Gamma_u} \boldsymbol{\epsilon}^{D^T}(\mathbf{v})\mathbf{C}^D\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \quad \forall \mathbf{v} \in H^1(\Omega)
\end{aligned} \tag{14}$$

$$\begin{aligned}
&-\int_{\Omega} q\left[\epsilon_V(\mathbf{u}) + \frac{p}{\kappa}\right]\mathrm{d}\Omega + \int_{\Gamma_u} q\mathbf{I}^T\mathbf{N}^T\mathbf{u}\,\mathrm{d}\Gamma \\
&\quad = \int_{\Gamma_u} q\mathbf{I}^T\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \quad \forall q \in L^2(\Omega)
\end{aligned} \tag{15}$$

where $H^1(\Omega)$ and $L^2(\Omega)$ are the first order Hilbert space and Lebesgue space of square integrable functions, respectively. κ is the bulk modulus; p is the pressure; ϵ_V and $\boldsymbol{\epsilon}^D$ are the volumetric and deviatoric components of the strain tensor and $\mathbf{C}^D$ is the matrix relating the deviatoric stress components to the deviatoric strain components.

We use the same approximation for the displacement field as in Eq. (5), and choose the following approximation for the pressure field

$$p(x,y) = \sum_{J=1}^{N} \sum_{n\in\mathcal{I}} h_{Jn}^p(x,y)p_{Jn} = \mathbf{H}_p(x,y)\mathbf{P} \tag{16}$$

where $\mathbf{P} = [p_{10}\ p_{11}\ p_{12}\cdots p_{Jn}\cdots]^T$ is the vector of nodal point unknowns corresponding to the pressure degrees of freedom. The shape function $h_{Jn}^p(x,y)$ at node J corre-

sponding to the nth degree of freedom is also generated using the partition of unity paradigm.

In [20] we have employed the numerical inf-sup test [3] to identify several combinations of these two spaces that result in a stable formulation.

Using the displacement and pressure approximations in Eqs. (14) and (15) we obtain the following discrete sets of equations corresponding to node I and degree of freedom m

$$\sum_{J=1}^{N}\sum_{n\in\mathcal{I}}\begin{bmatrix}\mathbf{K}_{uu_{ImJn}} & \mathbf{K}_{up_{ImJn}}\\ \mathbf{K}_{up_{ImJn}}^T & \mathbf{K}_{pp_{ImJn}}\end{bmatrix}\begin{Bmatrix}\boldsymbol{\alpha}_{Jn}\\ p_{Jn}\end{Bmatrix}=\begin{Bmatrix}\mathbf{f}_{Im}\\ \mathbf{0}\end{Bmatrix}+\hat{\mathbf{f}}_{Im} \tag{17}$$

where

$$\mathbf{K}_{uu_{ImJn}}=\int_{\Omega_I}\mathbf{B}_{Im}^{DT}\mathbf{C}^D\mathbf{B}_{Jn}^D\,\mathrm{d}\Omega \tag{18}$$

$$\mathbf{K}_{up_{ImJn}}=-\int_{\Omega_I}\mathbf{B}_{VIm}^T h_{Jn}^p\,\mathrm{d}\Omega \tag{19}$$

$$\mathbf{K}_{pp_{ImJn}}=-\frac{1}{\kappa}\int_{\Omega_I}h_{Im}^p h_{Jn}^p\,\mathrm{d}\Omega \tag{20}$$

and

$$\mathbf{f}_{Im}=\int_{\Omega_I}\mathbf{H}_{Im}\mathbf{f}^B\,\mathrm{d}\Omega \tag{21}$$

where $\Omega_I=\Omega\cap B(\mathbf{x}_I,r_I)$. If I is a node associated with an "interior sphere", then, of course

$$\hat{\mathbf{f}}_{Im}=\mathbf{0}$$

If the sphere corresponding to node I has a nonzero intercept on the Neumann boundary Γ_f, then

$$\hat{\mathbf{f}}_{Im}=\begin{cases}\int_{\Gamma_{f_I}}\mathbf{H}_{Im}\mathbf{f}^S\,\mathrm{d}\Gamma\\ \mathbf{0}\end{cases} \tag{22}$$

where $\Gamma_f=\cup_{\Gamma\in\mathcal{N}_f}\Gamma_{fI}$; $\mathcal{N}_f$ being the index set of such nodes.

On the other hand, if the sphere corresponding to node I has a nonzero intercept on the Dirichlet boundary Γ_u, then

$$\hat{\mathbf{f}}_{Im}=\sum_{J=1}^{N}\sum_{n\in\mathcal{I}}\begin{bmatrix}\tilde{\mathbf{K}}_{uu_{ImJn}} & \tilde{\mathbf{K}}_{up_{ImJn}}\\ \tilde{\mathbf{K}}_{up_{ImJn}}^T & \mathbf{0}\end{bmatrix}\begin{Bmatrix}\boldsymbol{\alpha}_{Jn}\\ p_{Jn}\end{Bmatrix}-\begin{Bmatrix}\tilde{\mathbf{f}}_{u_{Im}}\\ \tilde{\mathbf{f}}_{p_{Im}}\end{Bmatrix} \tag{23}$$

where

$$\tilde{\mathbf{K}}_{uu_{ImJn}}=\int_{\Gamma_{u_I}}\mathbf{H}_{Im}\mathbf{N}\mathbf{C}^D\mathbf{B}_{Jn}^D\,\mathrm{d}\Gamma+\int_{\Gamma_{u_I}}\mathbf{B}_{Im}^{DT}\mathbf{C}^D\mathbf{N}^T\mathbf{H}_{Jn}\,\mathrm{d}\Gamma \tag{24}$$

$$\tilde{\mathbf{K}}_{up_{ImJn}}=-\int_{\Gamma_{u_I}}\mathbf{H}_{Im}\mathbf{N}\mathbf{I}h_{Jn}^p\,\mathrm{d}\Gamma \tag{25}$$

$$\tilde{\mathbf{f}}_{u_{Im}}=\int_{\Gamma_{u_I}}\mathbf{B}_{Im}^{DT}\mathbf{C}^D\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \tag{26}$$

and

$$\tilde{\mathbf{f}}_{p_{Im}}=-\int_{\Gamma_{u_I}}h_{Im}^p\mathbf{I}^T\mathbf{N}^T\mathbf{u}^S\,\mathrm{d}\Gamma \tag{27}$$

where $\Gamma_u=\cup_{I\epsilon N_f}\Gamma_{uI}$; $\mathcal{N}_u$ being the index set of such nodes.

2.5 Numerical integration schemes

We review some of the schemes that we introduced in [1] for efficient computation of the terms in the local Galerkin weak form. We concentrate on two-dimensional domains and limit our discussion to "interior disks" (disks that have non zero intercepts with the domain boundary), "boundary sectors" (disks intercepted by the domain boundary) and the "lens" shaped regions of overlap of two disks.

For an "interior disk" we have implemented a piecewise midpoint quadrature rule (see Fig. 6a) by subdividing the disk using concentric circles and radial lines and evaluating the integral on each of the subdomains as the area of the subdomain multiplied by the integrand evaluated at the centroid of the subdomain. Hence, for the integral of a function $f(x,y)$ on a disk (Ω) of radius R_0 we use the following approximation

$$\iint_{\Omega}f(x,y)\,\mathrm{d}x\,\mathrm{d}y\simeq\sum_{i=1}^{n_\theta}\sum_{j=1}^{n_r}D_{ij}f(r_j\cos\,\theta_i,r_j\sin\,\theta_i) \tag{28}$$

where n_θ is the number of sectors in which the disk is subdivided and n_r is the number of subdivisions along each radius. Here

$$r_j\;=\frac{j^2-j+1/3}{j-1/2}\Delta r,\quad \theta_i=(i-1/2)\Delta\theta$$

where

$$\Delta r=\frac{R_0}{n_r},\quad \Delta\theta=\frac{2\pi}{n_\theta}$$

The weight

$$D_{ij}=\left(j-\frac{1}{2}\right)\Delta\theta(\Delta r)^2$$

is the area of the subdomain and is independent of the circumferential direction.

In [4] we categorized the boundary sectors into two major groups depending on the angle φ_0 that the radii joining the center of the disk to the two intercepts of the disk on Γ make interior to the domain:

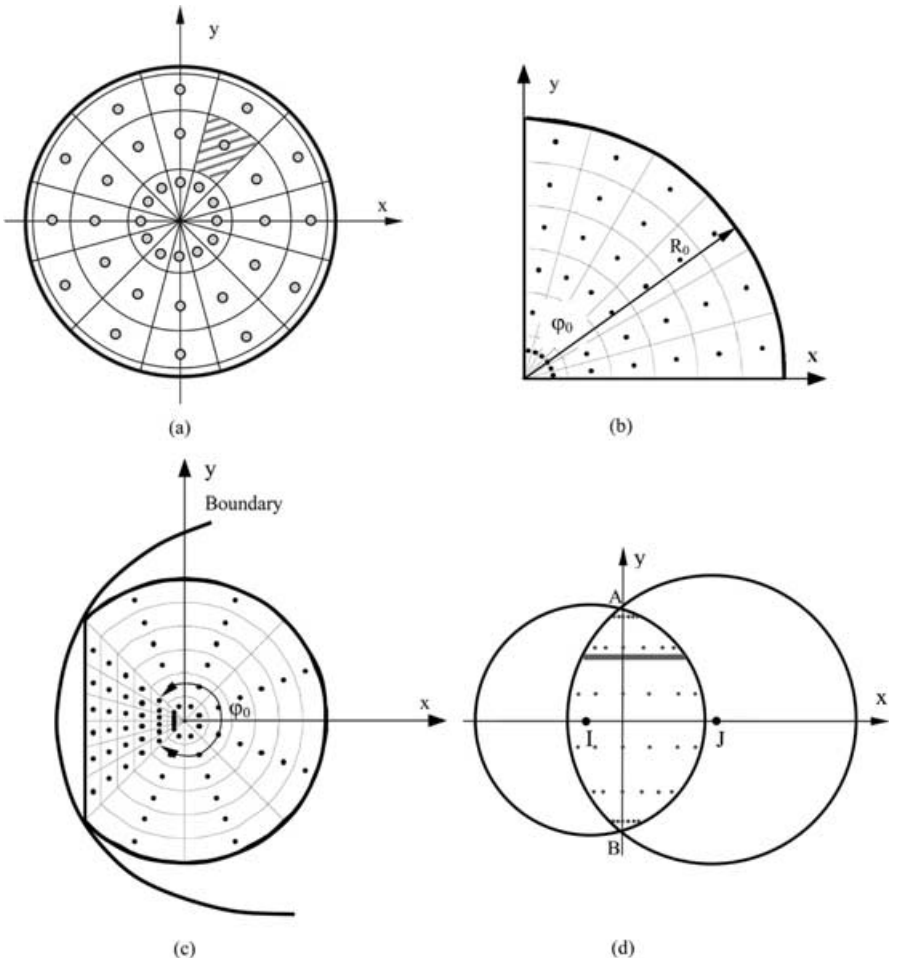


Fig. 6. Integration schemes on **a** an interior sphere; **b** a Type I boundary sphere;
c a Type II boundary sphere and **d** on the "lens" shaped overlap region of two spheres

Type I sector: $\varphi_0 \leq \pi$ (see Fig. 6b). The piece-wise midpoint quadrature rule (28) applies with the modification $\Delta\theta = \varphi_0/n_\theta$.

Type II sector: $\varphi_0 > \pi$ (see Fig. 6c). We may integrate over this type of boundary sector by first decomposing it into a sector for which the rule of the Type I sector can be used and a triangle.

For integrals defined on the "lens" shaped region of overlap of two disks (see Fig. 6d) the following rule has been developed. N_y integration points are chosen along the line AB corresponding to either a Gauss quadrature scheme or a piece-wise midpoint quadrature rule. The lens is subdivided into strips of width equal to the integration weights W_j^y. The area of the jth strip, A_j, may be computed analytically.

We may now write down an integration rule

$$\iint_{\Omega_{IJ}} f(x,y)\mathrm{d}x\,\mathrm{d}y \simeq \sum_{i=1}^{N_x}\sum_{j=1}^{N_y} D_{ij} f(x_i, y_j) \tag{29}$$

where

$$D_{ij} = \frac{A_j W_i^x}{l_j}$$

l_j is the length of a straight line drawn parallel to the x-axis through each integration point along AB, N_x is the number of integration points (corresponding to either a midpoint or a Gaussian quadrature rule) along each of these lines, and W_i^x is the weight associated with the ith integration point.

3
Point collocation based method of finite spheres

In this section we present a formulation of the method of finite spheres based on the point collocation technique for problems of linear elasticity. Numerical integration is circumvented as the governing differential equations are directly applied to the nodal points. Interpolation functions based on the moving least squares technique are used.

3.1
Governing equations

Let us consider the linear elasticity problem defined on a continuum $\Omega \in R^3$ with boundary Γ (see Fig. 7). The system of governing differential equations is

$$\boldsymbol{\partial}_\epsilon^T \boldsymbol{\tau}(\mathbf{u}) + \mathbf{f}^B(\mathbf{x}) = \mathbf{0} \quad \text{in } \Omega \tag{30}$$

with force (Neumann) boundary conditions

$$\mathbf{N}\boldsymbol{\tau}(\mathbf{u}) = \mathbf{f}^S(\mathbf{x}) \quad \text{on } \Gamma_f \tag{31}$$

and displacement (Dirichlet) boundary conditions

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}^S(\mathbf{x}) \quad \text{on } \Gamma_u \tag{32}$$

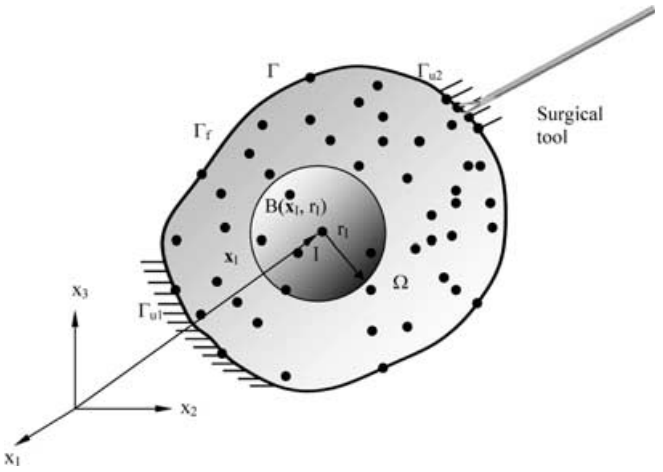


Fig. 7. The specialized boundary value problem corresponding to the problem of virtual surgery simulation. Homogeneous Dirichlet conditions are prescribed on portion Γ_{u1} of the boundary while nonhomogeneous Dirichlet boundary conditions are prescribed by the surgical tool interacting with the body on portion Γ_{u2} of the boundary

In Eqs. (30)–(32), $\mathbf{u}$ and $\boldsymbol{\tau}$ are the displacement and stress vectors, $\mathbf{f}^S$ is the prescribed traction vector on the Neumann boundary Γ_f, $\mathbf{u}^S$ is the vector of prescribed displacements on the Dirichlet boundary Γ_u (note that the domain boundary $\Gamma = \Gamma_f \cup \Gamma_u$ and $\Gamma_f \cap \Gamma_u = 0$), $\mathbf{f}^B$ is the body force vector (including inertia terms), $\boldsymbol{\partial}_\epsilon$ is a linear gradient operator, $\mathbf{N}$ is the matrix of direction cosine components of a unit normal to the domain boundary (positive outwards).

3.2
Point collocation

The point collocation technique [3] is a weighted residual scheme in which the displacement solution $\mathbf{u}$ is approximated by $\mathbf{u}_h$ and the governing partial differential equations are evaluated at the nodal points. The discrete set of equations may be written as

$$\left[\boldsymbol{\partial}_\epsilon^T \boldsymbol{\tau}(\mathbf{u}_h)\right]_{\mathbf{x}=\mathbf{x}_I} + \mathbf{f}^B(\mathbf{x}_I) = \mathbf{0} \quad \text{in } \Omega \tag{33}$$

with force (Neumann) boundary conditions

$$[\mathbf{N}\boldsymbol{\tau}(\mathbf{u}_h)]_{\mathbf{x}=\mathbf{x}_I} = \mathbf{f}^S(\mathbf{x}_I) \quad \text{on } \Gamma_f \tag{34}$$

and displacement (Dirichlet) boundary conditions

$$\mathbf{u}_h(\mathbf{x}_I) = \mathbf{u}^S \quad \text{on } \Gamma_u \tag{35}$$

where $\mathbf{x}_I$ is the position vector of node I.

3.3
Nodal interpolation

We choose the moving least squares functions as the trial functions, i.e.

$$\mathbf{u}_h(\mathbf{x}) = \sum_{J=1}^{N} \mathbf{H}_J(\mathbf{x})\boldsymbol{\alpha}_J = \mathbf{H}(\mathbf{x})\mathbf{U} \tag{36}$$

where N nodes are used for discretization and

$$\mathbf{U} = [\boldsymbol{\alpha}_1\ \boldsymbol{\alpha}_2\ \boldsymbol{\alpha}_3 \cdots]^{\mathrm{T}} \tag{37}$$

is the vector of nodal unknowns, and $\boldsymbol{\alpha}_J = [u^J\ v^J\ w^J]$ is the vector of nodal unknowns at node J (u^J, v^J and w^J are the nodal variables for the x, y and z direction displacements at node J). The nodal shape function matrix corresponding to the Jth node is

$$\mathbf{H}_J(\mathbf{x}) = \begin{bmatrix} h_J(\mathbf{x}) & 0 & 0 \\ 0 & h_J(\mathbf{x}) & 0 \\ 0 & 0 & h_J(\mathbf{x}) \end{bmatrix} \tag{38}$$

where

$$h_J(\mathbf{x}) = W_J(\mathbf{x})\mathbf{P}(\mathbf{x})^T\mathbf{A}^{-1}(\mathbf{x})\mathbf{P}(\mathbf{x}_J) \tag{39}$$

with

$$\mathbf{A}(\mathbf{x}) = \sum_{I=1}^{N} W_I(\mathbf{x})\mathbf{P}(\mathbf{x}_I)\mathbf{P}(\mathbf{x}_I)^T \tag{40}$$

The vector $\mathbf{P}(\mathbf{x})$ contains monomials (local basis functions) ensuring consistency up to a desired order. For a problem in R^3, for example, to ensure zeroth order consistency

$$\mathbf{P}(\mathbf{x})^T = [1]$$

and the resulting moving least squares shape functions in (39) are Shepard partition of unity functions. For first order consistency

$$\mathbf{P}(\mathbf{x})^T = [1, x, y, z]$$

and so on. The function $W_J(\mathbf{x})$ is a singular radial weighting function compactly supported on the sphere surrounding node J. It is well known (see [21]) that if such a weighting function is used for all the nodes in the domain then $h_J(\mathbf{x}_K) = \delta_{JK}\ \forall J, K$ (see [22] for a discussion on how to generate such functions efficiently). Therefore $\mathbf{U}$ in (37) is a vector of nodal displacements.

3.4
Discretized equations

Using (36) the discretized stress vector is

$$\boldsymbol{\tau}(\mathbf{x}) = \sum_{J=1}^{N} \mathbf{C}\mathbf{B}_J(\mathbf{x})\boldsymbol{\alpha}_J = \mathbf{C}\mathbf{B}(\mathbf{x})\mathbf{U}, \tag{41}$$

where the strain–displacement matrix $\mathbf{B}(\mathbf{x})$ in Eq. (41) is partitioned as

$$\mathbf{B}(\mathbf{x}) = [\mathbf{B}_1(\mathbf{x})\mathbf{B}_2(\mathbf{x}) \cdots \mathbf{B}_J(\mathbf{x}) \cdots]$$

where

$$\mathbf{B}_J(\mathbf{x}) = \boldsymbol{\partial}_\epsilon \mathbf{H}_J(\mathbf{x}) = \begin{bmatrix} \partial h_J/\partial x & 0 & 0 \\ 0 & \partial h_J/\partial y & 0 \\ 0 & 0 & \partial h_J/\partial z \\ \partial h_J/\partial y & \partial h_J/\partial x & 0 \\ 0 & \partial h_J/\partial z & \partial h_J/\partial y \\ \partial h_J/\partial z & 0 & \partial h_J/\partial x \end{bmatrix} \tag{42}$$

The discretized equations (33) and (34) may therefore be written as

$$\boldsymbol{\partial}_{\epsilon}^{T}\mathbf{CB}(\mathbf{x}_I)\mathbf{U} + \mathbf{f}^{B}(\mathbf{x}_I) = \mathbf{0} \quad \text{in } \Omega \tag{43}$$

and

$$\mathbf{NCB}(\mathbf{x}_I)\mathbf{U} = \mathbf{f}^{S}(\mathbf{x}_I) \quad \text{on } \Gamma_f \tag{44}$$

Equations (43), and (44) can be expressed in the compact form

$$\mathbf{KU} = \mathbf{f} \tag{45}$$

where $\mathbf{K}$ is the stiffness matrix (nonsymmetric but banded) and $\mathbf{f}$ is the vector containing nodal loads.

3.5
Fast point collocation for laparoscopic surgery simulation

We specialize the point collocation approach to the problem of laparoscopic surgical simulation. We consider a special boundary value problem where homogeneous Dirichlet conditions are prescribed on portion Γ_{u1} of the boundary and nonhomogeneous Dirichlet boundary conditions are prescribed on portion Γ_{u2} of the boundary where the virtual tool interacts with the organ (see Fig. 7), i.e.

$$\mathbf{u}(\mathbf{x}) = \mathbf{0} \quad \text{on } \Gamma_{u1} \tag{46}$$

and

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}_{\text{tool}} \quad \text{on } \Gamma_{u2} \tag{47}$$

where $\mathbf{u}_{\text{tool}}$ is the displacement field applied to the virtual organ through the tool (assumed known). Furthermore

$$\mathbf{f}^{S}(\mathbf{x}) = 0 \quad \text{on } \Gamma_f \tag{48}$$

and

$$\mathbf{f}^{B}(\mathbf{x}) = 0 \tag{49}$$

The key observation in the development of a fast solution procedure is that during the process of interaction of the surgical tool with the soft tissue, only the boundary conditions on Γ_{u2} change (assuming that the tissue is not ruptured).

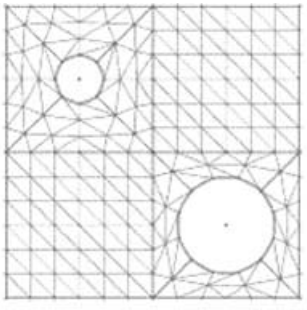
Let $\tilde{\mathbf{K}}$ be the stiffness matrix after incorporating the homogeneous Dirichlet conditions (46). Let $\tilde{\mathbf{C}}$ be the inverse of this matrix. This matrix can be partitioned as

$$\tilde{\mathbf{C}} = \begin{bmatrix} \tilde{\mathbf{C}}_{aa} & \tilde{\mathbf{C}}_{ab} \\ \tilde{\mathbf{C}}_{ba} & \tilde{\mathbf{C}}_{bb} \end{bmatrix} \tag{50}$$

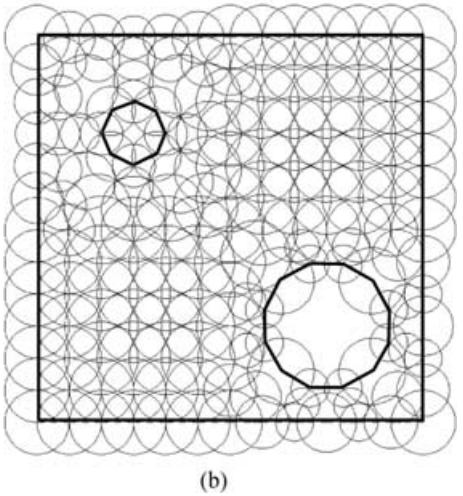
corresponding to a partitioning of the vector of nodal parameters as

$$\mathbf{U} = [\boldsymbol{\alpha}_{\text{tool}}\ \mathbf{U}_b]^{\mathrm{T}} \tag{51}$$

where $\mathbf{U}_b \in R^{3(N-m)}$ is the vector of unknown nodal displacements , $\boldsymbol{\alpha}_{\text{tool}} \in R^{3m}$ is the vector of *known* nodal displacements and m is the number of nodes on boundary Γ_{u2}. The reaction force vector at the tool $\mathbf{f}_{\text{tool}} \in R^{3m}$ is obtained from the relation

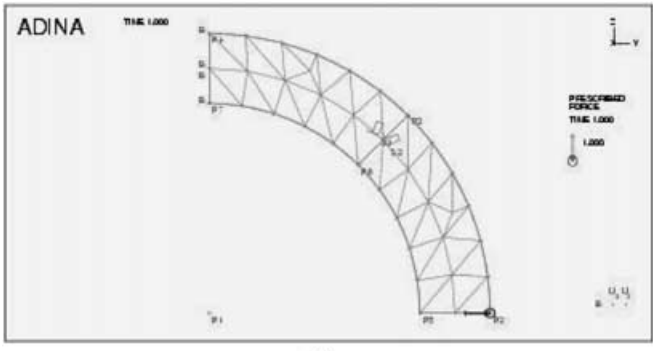


(a)

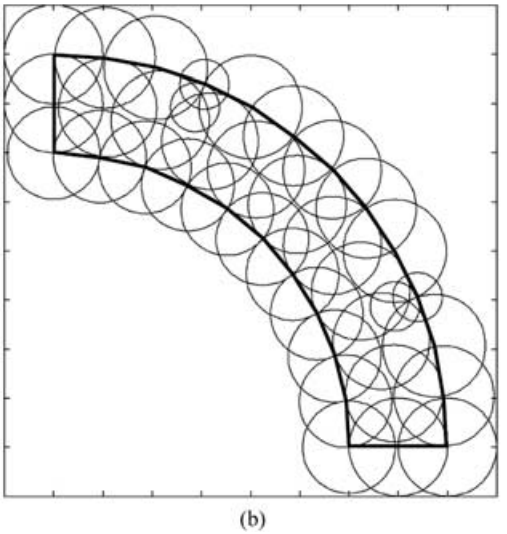


(b)

Fig. 8. A plate with two holes. The figure in **a** shows the finite element mesh generaed by ADINA. The figure **b** shows the open cover generated by 175 spheres



(a)



(b)

Fig. 9. **a** The finite element mesh generated on a quarter of a thin annulus is shown. The figure in **b** shows the open cover generated by 37 spheres

$$\mathbf{f}_{\text{tool}} = \left(\tilde{\mathbf{C}}_{aa}\right)^{-1}\boldsymbol{\alpha}_{\text{tool}} \tag{52}$$

and the unknown nodal displacements may be obtained as

$$\mathbf{U}_b = \tilde{\mathbf{C}}_{ba}\mathbf{f}_{\text{tool}} \tag{53}$$

If the matrix $\tilde{\mathbf{C}}$ is precomputed and stored, then the cost of computation of the tool reaction forces is $O(m^2)$ and of the displacement field is $O(m(N-m))$. Therefore, the overall computational complexity of such a procedure is $O(mN)$ which is essentially $O(N)$ when $m \ll N$.

4 Examples

Two examples demonstrating the techniques of generating a satisfactory cover using the ADINA input data file are presented in Fig. 8 and 9. In Fig. 8 a plate with two holes of

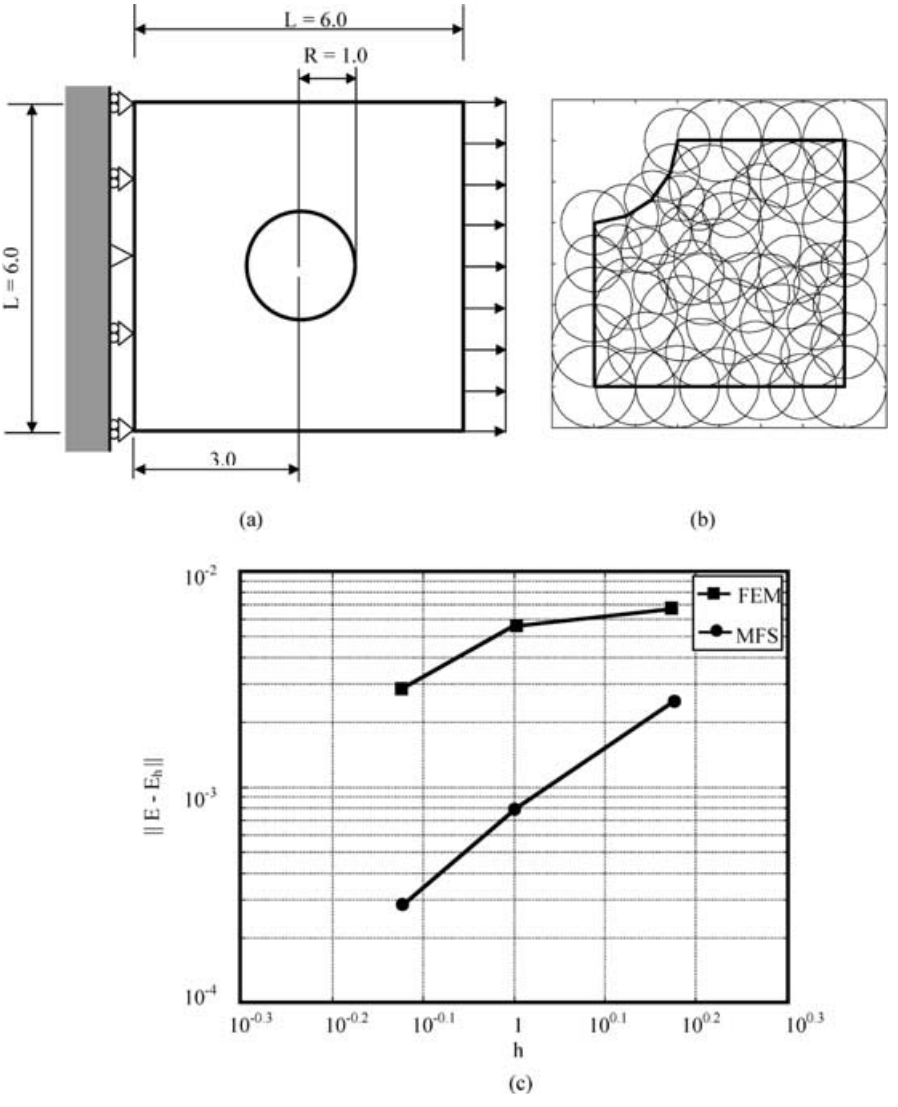


Fig. 10. **a** A square cantilever plate in plane strain conditions with a hole at the center. Uniform tensile loading is applied to one side. In **b** a quarter of the problem is considered and the covering of the domain using 50 spheres is shown. In **c** a comparison of the convergence in strain energy is shown when linear Galerkin finite elements as well as the method of finite spheres are used

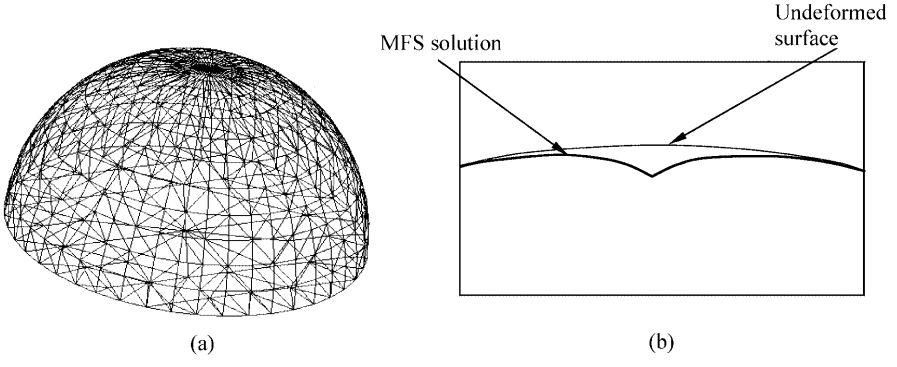


Fig. 11. The problem of a linear elastic hemisphere being indented by a tool-tip at the pole. The deformed surface, obtained using the point collocation -based method of finite spheres is shown in the vicinity of the tool tip together with the undeformed surface

different diameters is discretized using 175 spheres. Figure 9 shows one quarter of an annulus discretized using 37 spheres. The corresponding finite element meshes, generated using ADINA, are also shown.

In Fig. 10 we present the problem of a cantilever plate in plane strain conditions with a circular hole at its center. A uniformly distributed tensile loading is applied along one of the sides of the plate. Due to symmetry one quarter of the problem is discretized. A discretization using 50 spheres is shown in Fig. 10b. In Fig. 10c we compare the convergence in strain energy when the plate is discretized using Galerkin finite elements (linear) and the method of finite spheres (with linear local basis) when a h-type refinement is performed. It is observed that the method of finite spheres exhibits a superior convergence rate compared to the Galerkin finite element technique since approximation spaces exhibiting higher order continuity are used.

We have implemented the point collocation based method of finite spheres for real time simulation and display of deformation and tool tip reaction force for certain simple 3D geometries In Fig. 11b we show the displacement solution results obtained by applying vertical displacement to the pole of a hemisphere using the point collocation-based method of finite spheres (with linear local basis).

5
Concluding remarks

In this paper we have demonstrated how the data flow of a commercial finite element program, ADINA, can be used in the method of finite spheres in a general two-dimensional analysis. The superior node generation capabilities of a commercial finite element software can be harnessed and analysis results obtained using the method of finite spheres may now be directly compared with finite element results for exactly the same nodal configurations. This introduces spheres as a new class of "finite elements" and opens up the possibility of developing an integrated computer aided design/engineering environment where the user has the option of using either the spheres or the more traditional finite elements or a combination of both to solve a specific problem.

However, the techniques developed in this paper need to be generalized to three-dimensional domains before such a step may be implemented. While some of the techniques described here directly carry over to three-dimensions, new techniques for computing the intersections of spheres with complex three-dimensional boundaries need to be developed. Integration rules for three-dimensional computational subdomains are also required.

In addition, we have discussed a specialized version of the method of finite spheres, using point collocation, for rapid computation in multimodal virtual environments, specifically in laparoscopic surgery simulation. While the initial results are encouraging, techniques of simulating surgical cutting need to be developed. Moreover, a linear elastic model is inadequate since soft biological tissues are highly nonlinear, anisotropic and exhibit rate dependent properties. Efficient techniques of simulating such behavior need to be developed which allow the computations to be performed in real time.

References

1. **De S, Bathe KJ** (2001) The method of finite spheres with improved numerical integration. Comput. Struct. 79(22–25): 2183–2196
2. **De S, Bathe KJ** (2001) Towards an efficient meshless computational technique: the method of finite spheres. Eng. Comput. 18: 170–192
3. **Bathe KJ** (1996) Finite Element Procedures. Prentice Hall, Englewood Cliffs, NJ
4. **De S, Bathe KJ** (2000) The method of finite spheres. Comput. Mech. 25: 329–345
5. **De S, Kim J, Srinivasan MA** (2001) Virtual surgery simulation using a collocation-based method of finite spheres. Proceedings of the First MIT Conference on Computational Fluid and Solid Mechanics (KJ Bathe, ed), 2001, Elsevier.
6. **Massie T, Salisbury JK** (1994) The PHANTOM haptic interface: a device of probing virtual objects. Proc. ASME Winter Annual Meeting, pp. 295–301
7. **Salisbury JK, Srinivasan MA** (1997) Phantom-based haptic interaction with virtual objects. IEEE Comput. Graphics Appl. 17(5): 6–10
8. **Gibson SFF, Mirtich B** (1999) A survey of deformable modeling in computer graphics. MERL Report, pp. 1–31
9. **Cover S, Ezauerra N, O'Brien J** (1992) Interactively deformable models for surgery simulation. IEEE Comput. Graphics Appl. Magazine 68–75
10. **Terzopoulos D, Waters K** (1990) Physically based facial modeling, analysis and animation. J. Visualization Comput. Animation 1: 73–80
11. **Pieper S, Rosen J, Zeltzer D** (1992) Interactive graphics for plastic surgery: a task level analysis and implementation. Proc. Comput. Graphics pp. 127–134
12. **Bro-Nielsen M, Cotin, S** (1996) Real time volumetric deformable models for surgery simulation using finite elements and condensation. Proc. Comput. Graphics Forum, Eurographics 96, pp. 57–66
13. **Cotin S, Delingette H, Clement JM, Tassetti V, Marescaux J, Ayache N** (1996) Geometric and physical representations for a simulator of hepatic surgery. Proc. MMVR 4 Conf. pp. 139–151
14. **Cotin S, Delingette H, Ayache N** (1999) Real-time elastic deformations of soft tissue for surgery simulation. IEEE Trans. Visualization Comput. Graphics 5(1): 62–73.
15. **Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL** (1996) A finite point method in computational mechanics. Applications to convective transport and fluid flow. Int. J. Numer. Meth. Eng. 39: 3839–3866
16. **Yosida K** (1978) Functional Analysis. 5th edn. Springer-Verlag, Berlin Heidelberg
17. **Shepard D** (1968) A two-dimensional interpolation function for irregularly spaced data. Proc. 23rd Nat. Conf. ACM, pp. 517–524
18. **Klaas O, Shephard MS** (2000) Automatic generation of octree-based three-dimensional discretizations for partition of unity methods. Comput. Mech. 25: 296–304
19. **Macri M, De S, Shephard MS** Hierarchical tree-based discretization for the method of finite spheres. Comput. Struct. (in press)
20. **De S, Bathe KJ** (2001) Displacement/pressure mixed interpolation in the method of finite spheres. Int. J. Numer. Meth. Eng. 51: 275–292.
21. **Kaljevic I, Saigal S** (1997) An improved element free Galerkin formulation. Int. J. Numer. Meth. Eng. 40: 2953–2974
22. **Breitkopf P, Rassineux A, Touzot G, Villon P** (2000) Explicit form and efficient computation of MLS shape functions and their derivatives. Int. J. Numer. Meth. Eng. 48: 451–466