Proceedings of the 2006 IEEE/RSJ
International Conference on Intelligent Robots and Systems
October 9 - 15, 2006, Beijing, China

An Adaptive Meshless Computation Method for Design of Electromechanical Actuators

Qiang Li
G. W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0405
Gtg873k@mail.gatech.edu

Kok-Meng Lee*
G. W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0405
kokmeng.lee@ me.gatech.edu.

Abstract - **The ever increasing cost of energy and advance in permanent magnet technology have the incentives to develop geometrically compact and energy-efficient electro-magnetic (EM) actuators for robotic and automation applications. Design automation of these actuators often involves solving a magnetic field problem. This paper presents an adaptive meshless method (MLM) that inherits many advantages of FEM but needs no explicit mesh structure for design of EM actuators. Specifically, this paper offers a technique to estimate the distribution of numerical errors and a scheme automatically inserts additional nodes to improve computational accuracy and efficiency. Five examples are given; the first three are numerical examples, where exact solutions are available, provide a means to validate the adaptive MLM and evaluate its effectiveness against MLM with uniform node distribution. The other examples, where the magnetic forces are computed from the Lorenz's law, illustrate the use of adaptive MLM for the pole design of a three-DOF EM actuator. We also verify the results by comparing the computed forces against published experimental results.**

Index Terms - electromagnetic, adaptive meshless method, finite element, magnetic field, actuator design.

I. Introduction

Rapid growing interests in developing mobile and intelligent robots for applications in non-traditional industries (such as agriculture, food processing, medical service and entertainment industries) coupled with advance of low-cost high-coercive permanent magnet (PM) technology have motivated the development of novel electromechanical actuators that are geometrically compact and highly energy-efficient. The trends have been further accelerated by the ever increasing cost of energy. Design automation of the PM-based EM actuators involves solving a magnetic field problem.

With the advent of computational technologies, many engineering problems can be solved with numerical methods such as finite element method (FEM), boundary element method (BEM) and finite difference method (FDM). Among these, FEM has been most widely used as it can handle complicated geometry with the help of a mesh generation program. However computation accuracy of FEM depends on the quality of the mesh. Despite considerable effort has been devoted to improve the design of the mesh and the algorithm to generate it, the creation of a proper element structure remains a challenge; human involvement is still unavoidable for most of engineering analyses with FEM. Furthermore, the need to model both the dimensionally very small air-gaps where energy conversion takes place and the remaining electromagnetic structure presents a significant challenge.

Recently meshless methods (MLM), which inherit many advantages of FEM and yet, they need no explicit mesh structure to discretize geometry, have been proposed [1] and applied to some magnetic field computation [2]. The MLM method based on a similar theoretical framework as FEM has some unique advantages: firstly, it requires only scatter nodes (instead of elemental structure to discretize geometry, which significantly eases the preprocessing task. Secondly, it uses smooth shape functions to interpolate the field variables at a global level, which results in a smooth solution requiring no post-processing. Last but not least, it is an ideal method for adaptive computation since no element reconstruction is needed in the process of nodal insertion.

Most of earlier researches in MLM focus on proposing new methods for constructing basis functions. Little attention has been put on solving practical applications which exploit the advantages of MLM. Recently, some research efforts have been seen in solving two technical problems related to adaptive MLM. The first is to estimate computational error in MLM using methods such as residual technique in [3] and recovery technique in [4]. Although they are effective and efficient methods, these methods are mathematically difficult to derive and are complicated to apply in practice. The second is the development of a nodal insertion algorithm, which is needed to reconstruct integration cells after the nodes are inserted. Most adaptive MLM's use the background cell technique, which requires additional computation time particularly when the nodal distribution becomes irregular. Methods (such as quadtree technique) have been proposed to improve the efficiency of the reconstruction process [5, 6]. However, the additional computational load cannot be totally eliminated. In [7], a stabilized conforming nodal integration technique has been proposed to avoid the need for constructing background cells. This method has some successful applications in adaptive computation, for example [8] but its extension to three dimensional computations remains a challenge.

For these reasons, we offer an adaptive MLM for magnetic field computation for designs of electromagnetic actuators. The remainder of this paper offers the followings:

1) We offer an error estimation technique for adaptive MLM.

We extend the posteriori error estimation technique to MLM. This technique originally developed on the observation that the FEM results at certain locations (such as nodes) are

1-4244-0259-X/06/$20.00 ©2006 IEEE

more accurate than at other locations has achieved some successes in FEM [9, 10]. However, the basis function in MLM is, in general, not a polynomial; the posteriori error estimation technique developed for FEM can not be directly applied to MLM. In this paper, we present a modified error estimation built on two different support sizes of a basis function. As will be illustrated with a one-dimensional (1D) example, this modified error estimation characterizes the true error remarkably well, and its computation in MLM is simpler than in FEM.

2) We discuss practical issues related to the nodal insertion.

We present an automatic node insertion method based on a Voronoi plot technique along with the partition unity integration [11] scheme for obtaining the discretized system of weak-form formulated equations.

While the MLM does not need elements to perform numerical integration as in FEM, most of ML methods divide their computational domain into small numerical integration cells (called background cells) to ensure its accuracy. When the nodal density increases at a local area, it is desired to increase the density of the background integration cell at that area to ensure the accuracy, which makes MLM lose some of its advantages. For this reason, we introduce a different numerical integration scheme, called the partition unity integration, in which the density of integration cells simply increases with the number of nodes.

3) The adaptive ML computation is validated.

Three numerical examples, where exact solutions are available, are given to validate the computation of adaptive MLM. These examples illustrate the processes of error estimation and automatic node insertion. They also show the effectiveness of the adaptive MLM on the convergence by comparing the results against that those computed with uniform node distribution.

4) We illustrate design applications of EM actuators.

Two examples (Examples 4 and 5) illustrate the use of adaptive MLM for design of electromagnetic actuators with high coercive permanent magnets. In these examples, magnetic forces are computed using the Lorenz's force law. Example 4 compares the computational results against a set of published experimental results. Example 5 shows how the MLM can be used to analyze the pole design of a three DOF spherical motor to improve the torque-to-weigh performance.

II. Error Estimation in MLM

One of most common methods to improve the accuracy of the numerical approximation is to reduce the nodal space (or increase the density of nodes). The simplest way is to uniformly increase the nodal density in the whole computational domain. However, if the large numerical errors occurs only in certain local regions, this method is inefficient since extra nodes in small error regions do not help improve the overall computational accuracy but they would simply slow down the computational speed. Thus, it is desired to have an estimate of the overall error distribution of the computation so that additional nodes can be effectively inserted into large error regions accordingly.

The exact numerical error e can be defined as follows:

$$e(\mathbf{x}) = \Phi_e(\mathbf{x}) - \Phi_a(\mathbf{x}) \tag{1}$$

where $_e$ and $_a$ are the exact potential field distribution and the approximated solution using the MLM respectively. However, $_e$ is often unavailable in practice. Thus, a modified form is used to estimate the numerical error:

$$\hat{e}(\mathbf{x}) = \Phi_h(\mathbf{x}) - \Phi_l(\mathbf{x}) \tag{2}$$

where $_h$ and $_l$ are both numerical results but $_h$ is numerically more accurate than $_l$. As an example, $_l$ is a numerical solution obtained with $n\times n$ number of nodes, and $_h$ is a solution of $2n\times 2n$ nodes. However, it is desired that $_h$ can be computed without recalculating with the denser nodes for computational efficiency.

For the above reason, we present an alternative error estimation based on two different support sizes of a basis function to locate regions of large numerical errors for the adaptive MLM:

$$\tilde{e}(\mathbf{x}) = \sum_{i=1}^{n} \Psi_{i,d}(\mathbf{x})\Phi_{i,d} - \sum_{i=1}^{n} \Psi_{i,2d}(\mathbf{x})\Phi_{i,2d} \tag{3}$$

where $_{i,d}$ and $_{i,2d}$ denote the basis functions at the i^{th} node with a support size d and $2d$ respectively; $_{i,d}$ is the solution solved by initial computation; and $_{i,2d}$ is the fitted result using the basis function with a support size of $2d$**.** As an example, we include a commonly used MLM basis function (known as the reproducing kernel particle, or the RKP) basis function in the appendix. If the basis function is non-interpolating (as often the case in MLM), $_{i,2d}$ can be solved from the following system of linear equations:

$$\Phi_{2d} = \mathbf{E}^{-1}\Phi_d \tag{4}$$

where the elements of the matrix $\mathbf{E}$ are given as

$$\varepsilon_{i,j} = \Psi_{j,2d}(\mathbf{x}_i) \tag{5}$$

Once the error is estimated from (3), locations of large errors are identified as follows:

$$\forall \mathbf{x}_a : \tilde{e}(\mathbf{x}_a) > e_p \tag{6}$$

where x_a is the tested location; and e_p is a specified error threshold.

The rationale for (3) can be explained with the aid of Fig. 1, which compares two different support sizes of a RKP basis function. As shown in Fig. 1, the larger the support size the smoother is the basis function. In general, it is more difficult for the basis function with a larger support size to approximate a function with an abrupt change in their solution. Thus, regions of potential large errors can be located by comparing the approximations obtained the solutions of two different basis functions. Numerical experiments have confirmed this finding.

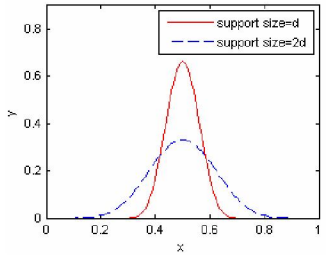


Fig. 1 Illustrating RKP basis function with two different support sizes

In this paper, the weak form equations are derived using Galerkin method in the MLM: First, the approximation (A.7) is substituted into the governing equation. Both sides of the equation are then multiplied by a test function. In WFF, both the test and trial functions are from the same functional space. Finally, the resulting equation is integrated over the entire domain.

Example 1: 1D problem illustrating the error estimation for MLM

We illustrate the method here using a 1D problem characterized by the 2nd order ordinary differential equation:

$$d^2y/dx^2 = f(x) \tag{7}$$

where $f(x) = -6x - e^{-(2x-1)^2/4\alpha^2}\left[2-(2x-1)^2/\alpha^2\right]/\alpha^2$. The boundary conditions are

$$y(0) = -e^{-1/4\alpha^2} \text{ and } y(1) = -1$$

where is a constant used to control the shape of solution. The exact solution is given by $y = -x^3 - e^{-(2x-1)/4\alpha^2}$.

To illustrate the error estimation, we solve (7) numerically using MLM with weak form formulation (WFF). The weak form equation is obtained by substituting the ML approximation

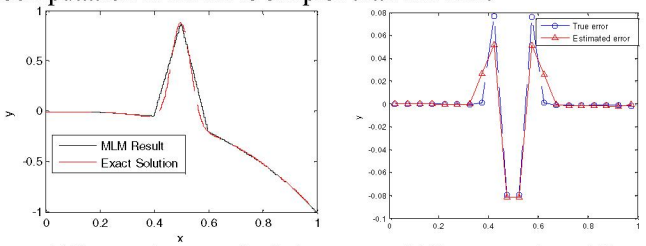
$$\hat{y}(x) = \sum_{i=1}^{n} \Psi_i(x) y_i$$

into (7) and integrating the result by parts, which yields

$$\int_0^1 \frac{d\Psi_j}{dx} \sum_{i=1}^{n} \frac{d\Psi_i}{dx} y_i dx = -\int_0^1 \Psi_j f(x) dx \tag{8}$$

For the purpose of inserting additional nodes, we compare the estimated error against the exact error with an emphasis on the error at the mid point between two adjacent nodes. As shown in Fig. 2(a), the solution has a high gradient region around *x=0.5*. The comparison in Fig. 2(b) shows that the results of the MLM (with a uniform distribution of 21 nodes) has a relatively large error around the high gradient region of *x=0.5*.

In order to insert additional nodes efficiently, the error estimation must identify this large error region faithfully with reasonable accuracy. As compared in Fig. 2(b), the estimated error characterizes the true error remarkably well, and its computation in MLM is simpler than in FEM.



(a) Exact and computed solutions (b) Exact vs estimated Error

Fig. 2 Comparison between exact and estimated errors

III. ADAPTIVE NODE INSERTION FOR MLM

Additional nodes can be inserted into the computational domain using the Voronoi plot [12] technique that constructs one Voronoi cell for each node.

Node Insertion Scheme:

An example Voronoi plot for a 2D computational domain is shown in Fig. 3, where the solid dots represent the nodes and the dash lines are boundaries of the Voronoi cells. As shown in Fig.3, a Voronoi cell is a polygon containing all the points closest to the node that it surrounds. The error at the vertexes of each Voronoi cell is computed from (2). If the error at a corner point satisfies criterion (6), a new node is created at that point as illustrated in Fig. 3. The three triangles at the corners of a Voronoi cell are example regions of large numerical errors.

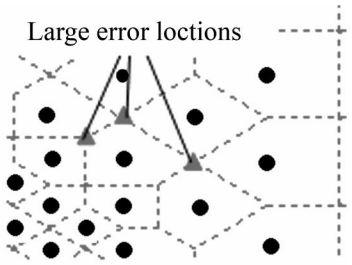


Fig. 3 Voronoi plot with 3 large error point

Support Size:

The support size of inserted node is calculated using (9) as the maximum distance from the node to its surrounding nodes whose voronoi cell is adjacent to this node:

$$r_i = a_p \cdot max\left(\left\|\mathbf{x}_j - \mathbf{x}_i\right\|\right) \tag{9}$$

where r_i is the support radius for i^{th} node; $\mathbf{x}_i$ is the coordinate of i^{th} node; $\mathbf{x}_j$ is the coordinate of j^{th} node. The voronoi cell of j^{th} node is adjacent to the Voronoi cell of i^{th} node. In (9); a_p is a constant coefficient, which is normally taken as 1~3. For the newly inserted node, the support size of its basis function must be chosen carefully considering the following trade-off:

1. The support radius must be sufficiently large to cover enough nodes for constructing the ML basis function. On the other hand, it is desired to localize the effect of the newly inserted nodes, and thus the support radius should be kept small.
2. Computational load increases as the support radius increases.

Partition unity integration:

The method of partition unity integration performs numerical integration based on the support of basis function. When a new node is inserted, a new integration cell is automatically created with no additional effort and thus, this numerical integration scheme is very suitable for adaptive computation.

Most of the basis functions used in MLM (including RKP basis function) have the partition unity property:

$$\sum_{i=1}^{n} \Psi_i(\mathbf{x}) = 1 \tag{10}$$

with which the integration for an arbitrary function $f(\mathbf{x})$ in the computational domain can be computed as follows:

$$\int_\Omega f(\mathbf{x}) d\mathbf{x} = \int_\Omega f(\mathbf{x}) \sum_{i=1}^{n} \Psi_i(\mathbf{x}) dx = \sum_{i=1}^{n} \int_\Omega f(\mathbf{x}) \Psi_i(\mathbf{x}) dx \tag{11}$$

where is the computational domain. To exclude points outside the computational domain, (11) is written such that the

integration is within the support domain S_i of i^{th} basis function:

$$\sum_{i=1}^{n}\int_{\Omega} f(\mathbf{x})\Psi_i(\mathbf{x})dx = \sum_{i=1}^{n}\int_{S_i} f(\mathbf{x})P(\mathbf{x})\Psi_i(\mathbf{x})dx \qquad (12)$$

where $$P(\mathbf{x}) = \begin{cases} 1 & when\ \mathbf{x}\in\Omega \\ 0 & when\ \mathbf{x}\notin\Omega \end{cases}$$

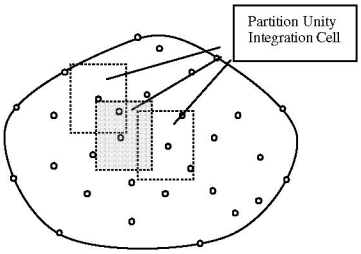


Fig.4 Partition Unity Integration cells

The global integration for the whole computational domain is divided into n sub-integration domains and performed upon the support domain of n basis functions. Because the support domain of the basis functions, in general, has a regular shape, the conventional numerical integration scheme such as Gaussian quadrature can be applied easily.

Example 2: Effect of adaptive node insertion on converging speed

Consider the 2D problem $$\nabla^2 u = \nabla^2 u_A(x,y) \qquad (13)$$

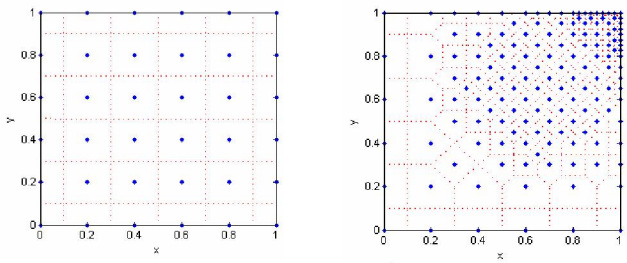
with the following boundary conditions:

$$u(x,0)=0\ ;\ u(x,1)=0\ ;u(0,y)=0\ ;\ u(1,y)=0 \qquad (14)$$

The exact solution is given by

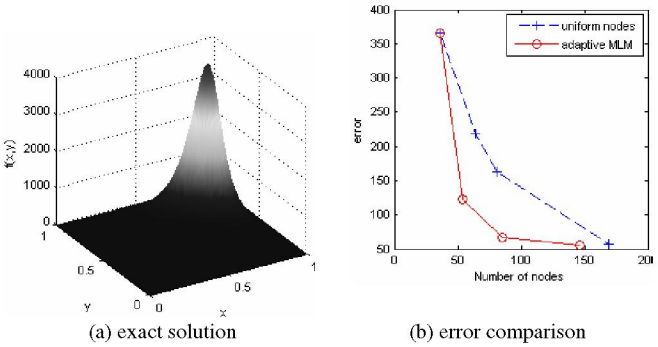
$$u_A(x,y) = 5x^2y^2(1-x)^2(1-y)^2(e^{10x^2}-1)(e^{10y^2}-1) \qquad (15)$$

We investigate the effects of adaptive node insertion on converging speed by comparing it against a typical weak-form-formulated MLM (with a globally uniform distribution of nodes). In other words, the nodes of the *uniform-node MLM* are increased uniformly in the computational domain while *adaptive MLM* increases its nodes according to the estimated error. Both methods start with an initial computation of 6×6 nodes. Three successive insertions are performed for each method. For the uniform-node MLM, the three successive node distributions are 8×8, 9×9 and 13×13. The node insertion of the adaptive MLM is automatically generated using the estimated error criterion (6) and is demonstrated in Fig.5.



(a) initial nodal distribution (36) (b) 3rd time nodal insertion (146)

Fig. 5 Process of adaptive nodal insertion



(a) exact solution (b) error comparison

Fig. 6 the exact solution of example and error comparison

For the purpose of comparison against the exact solution plotted in Fig. 6(a), we define the computation error as follows:

$$error = \sqrt{\iiint (u_{computed} - u_{exact})^2\, d\Omega} \qquad (16)$$

Since the computational time is directly proportional to the number of nodes, the comparison is made by plotting the computational errors versus the number of nodes in Fig. 6(b). The followings can be observed from the converging process of two methods:

1. Both methods tend to converge to the exact solution as the number of nodes increase. However, the adaptive MLM has a significantly higher converging rate.
2. The adaptive algorithm can effectively identify regions of large errors, which often occur around high gradient region.
3. The first two nodal insertions result in rapid error reduction as compared to the third insertion. As the number of nodal insertions increase, the error caused by highly irregular nodal distribution may gradually outweigh the benefit generated by additional nodes- this may hint that the number of nodal inserting iterations should not be too high in order to maintain the efficiency of algorithm.

Example 3: Handling the Discontinuity of Magnetic Field

The interest here is to demonstrate the use of adaptive node insertion to approximate discontinuities that often occur around the material interface of a magnetic circuit. For this purpose, we use the adaptive MLM to solve for the magnetic field intensity $\mathbf{H} = -\nabla\Phi$ around a cylindrical permanent magnet (uniformly magnetized along its axis, $\mathbf{M} = M_o\mathbf{i}_z$) in free space, where Φ is the magnetic scalar potential. In cylindrical coordinates, Φ can be solved from the Laplacian equation:

$$\nabla^2\Phi = \frac{1}{\rho}\frac{\partial}{\partial\rho}\left(\rho\frac{\partial\Phi}{\partial\rho}\right) + \frac{1}{\rho^2}\frac{\partial^2\Phi}{\partial\theta^2} + \frac{\partial^2\Phi}{\partial Z^2} = 0 \qquad (17)$$

where $\rho=r/R$, $Z=z/R$, and R is the radius of the magnets. For this axi-symmetric problem, $\partial\Phi/\partial\theta = 0$. The BC at the infinity far boundary is

$$\Phi_{\mathbf{x}\to\infty} = 0 \qquad (17a)$$

At the material interface, $\mathbf{H}$ is continuous along the tangential direction; and the flux density $\mathbf{B}$ is continuous along its normal. In terms of scalar potential functions,

$$\Phi_p = \Phi_q \qquad (17b)$$

$$\left(\mu_{rq}\nabla\Phi_q-\mu_{rp}\nabla\Phi_p\right)\bullet\mathbf{n}=\left(\mathbf{M}_q-\mathbf{M}_p\right)\bullet\mathbf{n} \qquad (17c)$$

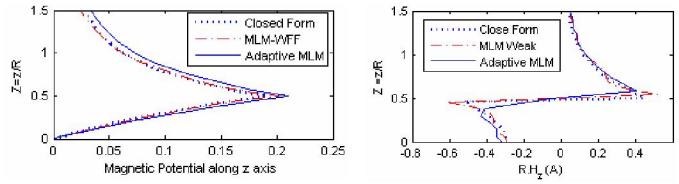
where the subscripts, p and q, denote as two different regions. In addition, for the symmetry

$$\partial\Phi/\partial r=0 \quad \text{at } r=0\,. \qquad (17d)$$

The weak form of this is shown in (18):

$$\oint_{\Gamma_m}\hat{\Psi}_j(\mathbf{M}_1-\mathbf{M}_2)\bullet\mathbf{n}d\Gamma-\oint_{\Omega}\mu_r\nabla\hat{\Psi}_j(\mathbf{x})\nabla\left(\sum_{i=1}^{n}\hat{\Psi}_i(\mathbf{x})\Phi_i\right)d\Omega=0 \qquad (18)$$

The magnetic scalar potential and field intensity along the z axis are obtained by the adaptive MLM and compared against the closed form solution in Figs. 7(a) and 7(b) respectively. The results of uniform-node MLM are obtained with 41×41 nodes. The adaptive MLM starts with a uniform nodal distribution of 221 (uniform 13x17) nodes and increases to 658 nodes after three successive insertions. Similar as in Example 2, all additional nodes are inserted into large gradient region which is the region around magnet pole in this case.



(a) magnetic scalar potential (b) magnetic field intensity

Fig. 7 Effect of adaptive node on handling of discontinuity

While the result of uniform-node MLM offers a very good overall prediction but exhibits some oscillations near the discontinuous interface. As compared in Fig. 7, the potential and field intensity computed by the adaptive MLM with only 658 nodes (less than half of the number of nodes used by the uniform–node MLM) match the closed form solution very well at the material interface. With a higher nodal concentration at the material interface, the adaptive MLM is able to approximate the discontinuity satisfactorily with a continuous basis function. This suggests that the adaptive MLM is a good alternative for solving problem with field discontinuity.

IV. POLE DESIGN OF ELECTROMAGNETIC (EM) ACTUATORS

An important step in designing EM actuators is to predict the field interaction between the stator and rotor poles. The following two examples illustrate the application of the field results obtained using the adaptive MLM in Example 3 for practical design of EM actuators. Example 4 verifies the computational results against published experimental data. Example 5 analyses the pole design of a three DOF spherical motor for improving the torque-to-weigh performance.

Example 4: Force between EM and PM

This example is selected from one of T.E.A.M. problems [13], where the experimental setup is shown in Fig. 8. The forces between coil and magnet are computed from the Lorenz's law for two configurations with the dimensions listed in Table 2. Once the magnetic field is computed, the force on the current volume can be obtained by integrating the force density exerted on the current carrying conductor by its interaction with the magnetic field:

$$\mathbf{F}=\int_V(\mathbf{J}\times\mathbf{B})rdrd\theta dz \qquad (19)$$

where $\mathbf{J}$ is current density; and V is the volume of the current conductor. Fig. 9 compares the computational result with experimental result. The computed restoring force matches the experimental results very well while the computed axial force is slightly larger than measured force.

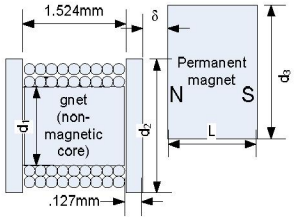
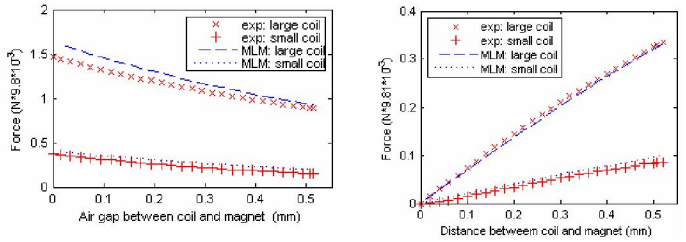


Fig. 8 Experiment configuration

Table 1: Dimensions of Example 4

Configurations	Large	Small
d_1 (mm)	3.048	1.524
d_2 (mm)	3.962	3.175
d_3 (mm)	2.998	1.6
L (mm)	1.6	0.8128
Coil res. ( )	57	32
Wire length (m)	3	1.68

Samarium-Cobalt magnet; μ_0M_0=1.02 T



(a) Axial force (b) Restoring force

Fig. 9 Comparison between computed and experimental results

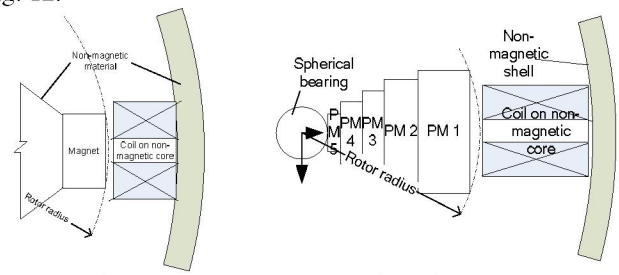
Table 2 Parameters used in simulation

Design	Rotor radius, mm	Stator Coil OD×ID×L (mm)	# of turns	PM rotor pole OD×L (mm)
1	37.5mm	19.05×9.53×25.4	1050	12.7×12.7
2	46.5	18×4×27	900	25×10, 20×5, 16×6, 12×3, 8×3

Air gap = .5mm; Magnetization μ_0M_0=1.27T

Example 5: Pole design of a Three-DOF spherical actuator

Figure 10 compares two example pole designs [14, 15] for a spherical motor capable of providing three degrees of freedom (DOF) motion in a single joint. The geometry and layout of the magnetic poles have a significant influence on the torque performance of the motor. The rotor of Design #1 consists of two rows of 8 small PM's whereas Design #2 uses one row of 8 large PM's. Detailed geometries of the two pole designs are given in Table 2. Fig. 12 compares the torque output per unit radius computed using the adaptive MLM with Lorenz Law for a single EM-PM pole-pair. The effects of pole layout on the torque profile for Design #1 are plotted in Fig. 12.



(a) Design 1 [22] (b) Design 2 [23]

Fig. 10 Comparison of pole designs (not to scale)

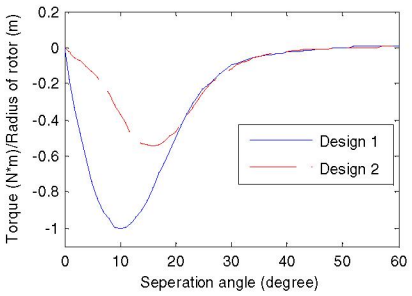
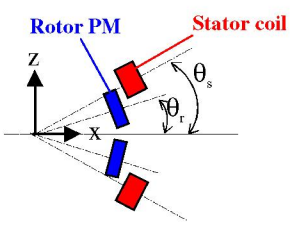
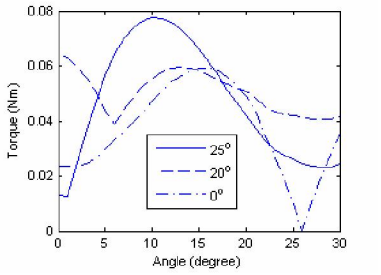


Fig. 11 Comparison of torque per unit radius



For θ_r=0°, only one PM is used, which has the same length but double the volume.

(a) simulated pole geometry



(b) Torque for different θ_r (θ_s=26°)

Fig. 12 Effect of pole layout on torque generated

As compared in Fig. 12, the torque-to-radius ratio for Design #1 is significantly higher although Design #2 uses a much larger PM rotor pole for a similar input power. In addition, the compact pole geometry in Design #2 allows for a larger number of rotor poles to be used and hence further improve the torque performance for a given volume and input power. These results illustrate how the adaptive MLM can be effectively used to analyse the effects of pole design on the torque performance.

V. Conclusion

A relative complete adaptive computational method for MLM has been presented and illustrated with five examples, which offers a solution to overcome two technical difficulties associated with MLM; namely, error estimation and nodal insertion. The computed method has been validated by comparing against exact solutions using three examples. Additionally, we compare the electromagnetic forces computed using Lorenz's law with the field results predicted by the adaptive MLM against published experimental results. These comparisons show excellent agreement. The effectiveness of the method has been evaluated by comparing the converging speed against MLM with uniform distributed nodes. Our results show that the method can faithfully locate large error regions, automatically insert nodes to these regions without human's involvement, and improve the computational efficiency significantly.

References

1 Babuska, I. and J. M. Melenk, (1997). "The Partition of Unity Method," *I. J. for Num. Methods in Eng.*, vol. 40, pp. 727-758.

2 Liang, X., Z. Zhiwei, B. Shanker, and L. Udpa, (2004). "Element-free Galerkin method for static and quasi-static electromagnetic field computation," *IEEE Trans. on Magnetics*, vol. 40, pp. 12.

3 Park, S.-H., K.-C. Kwon, and S.-K. Youn, (2003). "A posteriori error estimates and an adaptive scheme of least-squares meshfree method," *I. J. for Num. Methods in Eng.*, vol. 58, pp. 1213.

4 Rossi, R. and M. K. Alves, (2005). "An h-adaptive modified element-free Galerkin method," *European J. of Mechanics, A/Solids*, vol. 24, pp. 782.

5 Rabczuk, T. and T. Belytschko, (2005). "Adaptivity for structured meshfree particle methods in 2D and 3D," *I. J. for Num. Methods in Eng.*, vol. 63, pp. 1559.

6 Tabarraei, A. and N. Sukumar, (2005). "Adaptive computations on conforming quadtree meshes," *Finite Elements in Analysis and Design*, vol. 41, pp. 686.

7 Chen, J.-S., C.-T. Wu, S. Yoon, and Y. You, (2001). "Stabilized conforming nodal integration for Galerkin mesh-free methods," *I. J. for Num. Methods in Eng.*, vol. 50, pp. 435.

8 You, Y., J. S. Chen, and H. Lu, (2003). "Filters, reproducing kernel, and adaptive meshfree method," *Computational Mechanics*, vol. 31, pp. 316.

9 Zienkiewicz, O. C. and J. Z. Zhu, (1992). "Superconvergent patch recovery and a posteriori error estimates. Part 2: error estimates and adaptivity," *I. J. for Num. Methods in Eng.*, vol. 33, pp. 1365.

10 Zienkiewicz, O. C. and J. Z. Zhu, (1992). "Superconvergent patch recovery and a posteriori error estimates. Part 1: the recovery technique," *I. J. for Num. Methods in Eng.*, vol. 33, pp. 1331.

11 Carpinteri, A., G. Ferro, and G. Ventura, (2002). "The partition of unity quadrature in meshless methods," *I. J. for Num. Methods in Eng.*, vol. 54, pp. 987.

12 Barber, C. B., D. P. Dobkin, and H. Huhdanpaa, (1996). "The Quickhull algorithm for convex hulls," *ACM Trans. on Mathematical Software*, vol. 22, pp. 469.

13 Bastos, N. I. J. P. A. "Forces in Permanent Magnets Team Workshop Problem 23," *http://www.compumag.co.uk/team.html*.

14 Lee, K.-M. and H. Son, 2005. "Torque model for design and control of a spherical wheel motor," Monterey, CA, United States.

15 Yan, L., I.-M. CHen, C. K. Lim, G. Yang, W. Lin, and K.-M. Lee, 2002. "Torque Modeling of a DC Spherical Actuator Based on Lorentz Force Law," presented at International Conference on Control, Automation, Robotics and Vision, Singapore.

APPENDIX

Reproducing kernel particle (RKP) basis function

The RKP basis function can be expressed as

$$\Psi_i(\mathbf{x}) = C(\mathbf{x};\mathbf{x}-\mathbf{x}_i)\Lambda(\|\mathbf{x}-\mathbf{x}_i\|/d) \tag{A.1}$$

where $\Lambda(\|\mathbf{x}-\mathbf{x}_i\|/d)$ is a kernel (or weight) function centered at $\mathbf{x}_i$; the support size d is a design parameter that influences the effective region of the kernel function.and $C(\mathbf{x};\mathbf{x}-\mathbf{x}_i)$ is a set of enrichment functions that vary with the location of approximation $\mathbf{x}$. The following cubic B-spline function [14] is chosen for the kernel function:

$$\Lambda(\mathbf{x}-\mathbf{x}_i) = \begin{cases} 2/3-4p^2+4p^3 & for\ 0\le p\le 1/2 \\ 4(1-3p+3p^2-p^3)/3 & for\ 1/2\le p\le 1 \\ 0 & for\ p\ge 1 \end{cases} \tag{A.2}$$

where $p = \|(\mathbf{x}-\mathbf{x}_i)\|/d$; The function $C(\mathbf{x};\mathbf{x}-\mathbf{x}_i)$ is given by [14]:

$$C(\mathbf{x};\mathbf{x}-\mathbf{x}_i) = \mathbf{h}^{\mathrm{T}}(0)\mathbf{P}^{-1}(\mathbf{x})\mathbf{h}(\mathbf{x}-\mathbf{x}_i) \tag{A.3}$$

where $\mathbf{h}^{\mathrm{T}}(\mathbf{x}-\mathbf{x}_i) = \begin{bmatrix} 1 & (\mathbf{x}_1-\mathbf{x}_{i1}) & \cdots & (\mathbf{x}_n-\mathbf{x}_{in})^n \end{bmatrix}$; and $\mathbf{h}^{\mathrm{T}}(0) = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix}$;

$$\mathbf{P}(\mathbf{x}) = \mathbf{P}_o(\mathbf{x}) + \sum_{i=1}^{n_a} \mathbf{h}(\mathbf{x}-\mathbf{x}_i)\mathbf{h}^{\mathrm{T}}(\mathbf{x}-\mathbf{x}_i)\Lambda(\mathbf{x}-\mathbf{x}_i) \tag{A.4}$$

where n_a is the number of newly added nodes.

To impose the boundary conditions in WFF, a modified basis function $\hat{\Psi}_i(\mathbf{x})$ that is used:

$$\hat{\Psi}_i(\mathbf{x}) = \sum_{j=1}^{n} \Psi_i(\mathbf{x}) L_{ij}^{-\mathrm{T}} \tag{A.5}$$

where the element L_{IJ} is defined by

$$L_{ij} = \Psi_i(x_J) \tag{A.6}$$

such that

$$\tilde{\Phi}(\mathbf{x}) = \sum_{i=1}^{n} \hat{\Psi}_i(\mathbf{x})\Phi_i \tag{A.7}$$