Applied Science and Innovative Research
ISSN 2474-4972 (Print) ISSN 2474-4980 (Online)
Vol. 4, No. 4, 2020
www.scholink.org/ojs/index.php/asir

Original Paper

Parallel Efficient Mesh Deformation Method Based On Support Vector Regression

Haixiang Liao[1] & Xiang Gao[1*]

[1] College of Computer, National University of Defense Technology, Changsha, China

[*] Correspondence, Xiang Gao, College of Computer, National University of Defense Technology, Changsha, China

Received: November 10, 2020 Accepted: November 26, 2020 Online Published: November 27, 2020

doi:10.22158/asir.v4n4p54 URL: http://dx.doi.org/10.22158/asir.v4n4p54

Abstract

Mesh deformation method is widely used in unsteady numerical simulations involving moving boundaries. This kind of method redistributes the position of grid points in accordance with the movement of the computational domain without changing their connectivity relations. In this paper, we present a parallel mesh deformation method based on the support vector machine regression (SVR). In each time step, the proposed method first trains three SVRs by the coordinates of the boundary points and their known displacements in each direction, and then predicts the displacements of the internal points using the SVRs. After deforming the mesh, the dual-time step flow solver is used to solve the governing equations. To ensure the consistency of the method running in parallel, the training part of the method is executed with all global boundary points in each decomposed domain. Therefore, each CPU needs to maintain a copy of the entire boundary points via a point-to-point communication. The internal evaluation of the method is predicted separately in each decomposed domain without any data dependency. An oscillatory and transient pitching airfoil case is simulated to demonstrate the applicability of the proposed mesh deformation method, and its parallel efficiency is over 60% with 64 cores.

Keywords

Mesh deformation, Support vector machine, Radial basis function, Parallelization, Computational fluid dynamics

1. Introduction

With the rapid development of computer technology, numerical simulations involving moving boundaries have been widely applied in various fields like computational fluid dynamics (CFD), and

the size of the required grids shows explosive growth along with different cell types. Therefore, a robust and efficient mesh deformation approach, which relocating the mesh points to fit the new computational domain without changing topological relations, is extremely important to speed up the complex simulation in parallel situations. An example of the mesh deformation method deforming an unstructured mesh is illustrated in Figure 1.

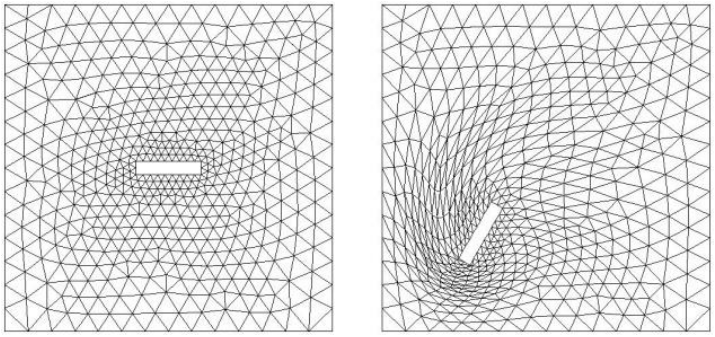

Figure 1. The Schematic Meshes that before and after Moving the Block Using Mesh Deformation

There are two categories of deforming strategies in literature. The first utilizes the connectivity information of the mesh by analogizing the edges as springs or solid body elasticity (De Boer, Van Der Schoot, & Bijl, 2007). These methods solve the mesh motion by reaching a new equilibrium of the spring system or through a proposed differential equation using certain boundary conditions. In hence, they need to solve an equation whose size equals to the number of grid points, which is computationally burdensome for large-scale simulations. The other connectivity free strategy moves each grid point independently according to its coordinates or distance to the boundary, and its parallelization is easily implemented. In recent years, several connectivity free mesh deformation methods have been reported in literature. The radial basis function (RBF) interpolation method is robust and widely used in the community (Rendall & Allen, 2009; Sheng & Allen, 2003). It interpolates an approximation function describing the displacement of the entire computational domain in each coordinate direction. Therefore, substituting the coordinates of the interior point into the function will get its offset. An equation system with the scale of the boundary nodes needs to be solved. It is still expensive for large-scale simulations. Although a data reduction schemes (Rendall & Allen, 2009) based on minimizing an error function is proposed to reduce the selection of boundary nodes, this iterative algorithm needs to solve dense linear equations with increasing size, which is difficult to solve using standard iterative techniques.

In this study, we present a new parallel mesh deformation method based on the support vector machine (Pal, Basak, & Patranabis, 2007). The final form of this machine learning algorithm is similar to the aforementioned RBF method. There are two steps in our approach. The first step is to compute a regression function formed by a summation of kernel functions over boundary points, but the SVR

method selects key boundary points automatically, and the function coefficients are obtained by solving a simple quadratic programming problem instead of a dense linear system of equations (Burges, 1998). The second step is to calculate the displacements of internal points individually according to the trained function. Since it avoids computationally intensive operations and preserves the parallel ability, this approach is more efficient than the RBF interpolation method. To demonstrate the deforming capability and parallel efficiency of the SVR method, a typical pitching airfoil case is tested with the inviscid and compressible flow.

2. Methodology

This section first describes the basic methodology of the SVR method; afterwards the parallelization and its global procedure applied in real physical problems are discussed.

2.1 Support Vector Machine Regression

The support vector machine was originally developed by Vapnik et al. based on statistical learning theory since 1979 (Pal, Basak, & Patranabis, 2007; Cherkassky & Ma, 2004). In recent thirty years, SVR has been successfully applied in various fields such as face detection, time series and financial prediction. There are two main categories of SVR based on classification and regression (Li, Wang, Gu, & Ling, 2015; Li, Yin, Lu, Kong, Zhang, & Liu, 2013). For the purpose of mesh deformation, SVR regression can be utilized to predict the displacements of grid points. The idea of SVR regression is to calculate a linear function in a high-dimensional feature space mapping from the input space via a nonlinear kernel function. The resulting function produced by SVR regression only depends on a subset of the training data, because it ignores the training data that lie within a threshold ε to the prediction function, as shown in Figure 2.

Supposing the training data $\{(\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n)\} \subset \mathbb{R}^d \times \mathbb{R}$, where $\mathbb{R}^{\mathrm{d}}$ denotes the space of input data. The case of linear function $f(\vec{x})$ can be described in the form as

$$f(\vec{x}) = \langle \vec{\omega}, \vec{x} \rangle + b \; with \; \vec{\omega} \in \mathbb{R}^d, b \in \mathbb{R}$$

where $\langle .,. \rangle$ denotes the dot product in $\mathbb{R}^d$. The goal of SVR regression is to find a function that has at most ε deviation from y_i for all the training data, and at the same time make the function as flat as possible. Flatness means to minimize the Euclidean norm, therefore this can be written as the following convex optimization problem:

$$min \frac{1}{2} \parallel \vec{\omega} \parallel^2 \text{ subject to } \begin{cases} y_i - \langle \vec{\omega}, \vec{x}_i \rangle - b \leq \varepsilon \\ \langle \vec{\omega}, \vec{x}_i \rangle + b - y_i \leq \varepsilon \end{cases}$$

This problem is solvable if f actually exists and approximates all data pairs with ε precision. Figure 2 graphically demonstrates the regression solution. In the mesh deformation, we assume all training data lie in/on the ε-insensitive tube around f without exception.

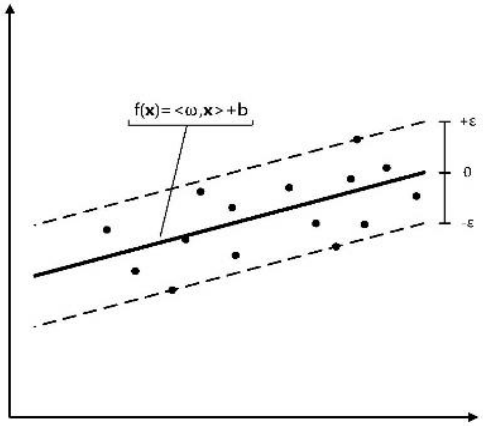

Figure 2. Linear SVR Model

SVR can be easily extended to nonlinear cases by mapping the input space into higher dimensional feature space, by using the kernel function k defined as a linear dot product of the nonlinear mapping. Furthermore, the optimization problem can be dualized by utilizing Lagrange multipliers (Burges, 1998), and the resultant function can be rewritten as follows:

$$f(\vec{x}) = \sum_{i=1}^{n} (\alpha_i - \hat{\alpha}_i) k(\vec{x}_i, \vec{x}) + b$$

The coefficients $\alpha_i, \hat{\alpha}_i$ can be efficiently calculated by solving the special dual problem using the sequential minimal optimization (SMO) algorithm (Platt, 1998).

In this work, we applied the Gaussian kernel function to map the input data:

$$e^{-\frac{\|\vec{x}_i - \vec{x}_j\|^2}{2\sigma^2}} \qquad \sigma > 0$$

where σ is the kernel width parameter that controls the radial scope of the function. With the increase of σ, the faster the Gaussian function decays. In the mesh deformation, we could set the kernel width close to the size of the deformable part of the shape (e.g., the chord length of the airfoil), so the displacements of the control points mainly affect the appropriate local domain. The coordinates of boundary points and their known displacements of the next step are used as training samples in mesh deformation. After training the SVRs (one coordinate direction for an SVR), the displacements of internal points can be computed individually just like the RBF method. It should be noticed that there is no need to pre-scale the data to [0, 1] range as what SVR usually does, because the dimension of each component of the grid coordinates is the same, and the normalization may degrade the data accuracy.

2.2 Global Procedure Coupled with CFD

While the mesh deformation technique is proposed, it needs to be coupled with CFD calculation for moving boundary numerical simulations. The displacements of the moving boundary points are usually obtained by boundary movement equations. These equations are independently predefined or coupled with flow variables. The global procedure of the parallel mesh deformation solver is presented in Figure 3. When the solver runs in parallel, it needs to decompose and distribute the computational

domain to each CPU in the preprocess stage. In each time step, the SVR method is first applied to deform each sub-mesh according to the same SVRs. Because the topological relationship among mesh cells is not changed, there is no need to interpolate new flow quantities in mesh cells. Only geometric information like cell volumes and face areas is recalculated. After generating the new mesh, the "CFD time iteration" section is executed. Finally, once the unsteady simulation is terminated, it reconstructs the mesh and flow fields as a whole.

To ensure the deformation in each subdomain is consistent, we use the same training data in each CPU to form the same SVRs. Therefore, each CPU needs to maintain a copy of the entire boundary points and their displacements via a point-to-point communication. After each CPU computes the fitting function, the solver evaluates the displacements of internal points in each decomposed domain simultaneously. In hence, the proposed mesh deformation solver is successful extended to parallel environment coupling with CFD solvers.

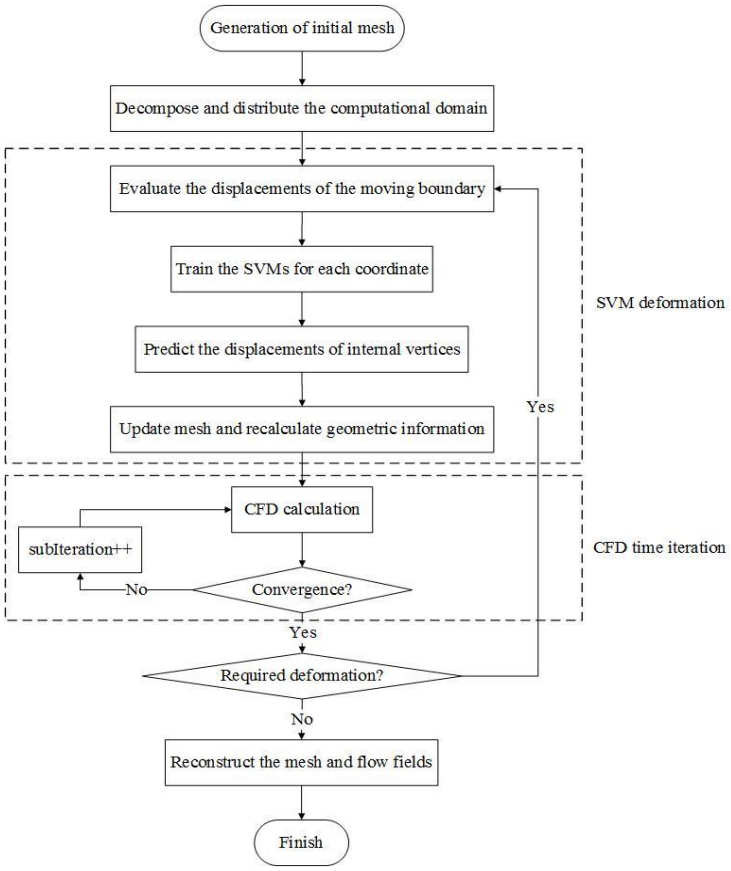

Figure 3. The Global Procedure of the SVR Mesh Deformation Method

3. Results

The new parallel mesh deformation method is tested with a calculation of inviscid flow around a pitching NACA0012 airfoil case to demonstrate its deforming applicability and efficiency. The case is tested on a high-performance cluster with several nodes. Each node has two Intel Xeon E5-2692 v2 CPUs, which in all has 24 cores at 2.20 GHz, and 64 GB memory.

The initial computational mesh is shown in Figure 4. The mesh has 40,400 internal points, 800 boundary points and 20,400 quadrilateral elements, and extends 75 times chord length from the airfoil with a circular outer boundary. The unsteady calculations are performed for the airfoil pitching harmonically about the 1/4 chord with the following formula (Landon, 1982):

$$\alpha = \alpha_0 + \alpha_m \sin(kt)$$

where the amplitude of $\alpha_m = 2.51^{\circ}$, the reduced frequency k based on semichord and far-field Mach number is 0.0814, the far-field Mach number $M_\infty = 0.755$, and $\alpha_0 = 0.016^{\circ}$. After transformating the above dimensionless formula into dimensional, one oscillatory cycle takes about 0.147 s.

The flow solver uses the linear least-squares reconstruction technique, the arbitrary Lagrangian-Eulerian (ALE) formulation of HLLC flux function and implicit dual-time step approach with the LU-SGS method (Daude, Galon, Gao, & Blaud, 2014; Zhang & Wang, 2004). The flow starts with the steady results at $\alpha = \alpha_0$, and the step size is chosen as $\Delta t = 4 \times 10^{-4}$ s (about 1/368 cycle).

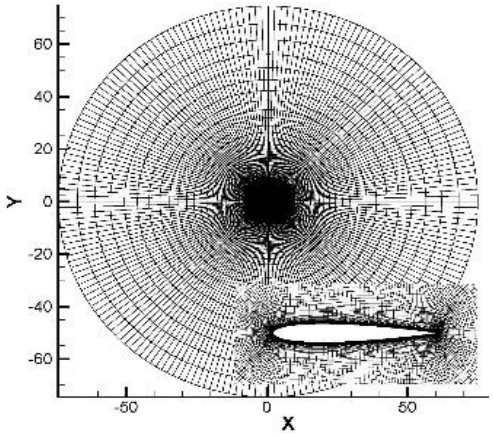

Figure 4. Initial Mesh of the NACA0012 Airfoil with a Detailed View

The initial pressure field calculated with steady condition is presented in Figure 5. The computational mesh at the maximum amplitude deformed by the SVR method is illustrated in Figure 6. We can see that the boundary mesh still move together with the airfoil and its mesh quality has been preserved well during the deformation. In this 2D case, the average number of support vectors (or called control points) in each direction is about 144 selected by the SVR method in each time step. This is about 18% of all the boundary points. Therefore, it is much more efficient compared with the RBF method using all the boundary points.

In this case, three cycles of motion are computed to obtain a periodic solution. Comparisons of calculated and experimental (Landon, 1982) normal and moment coefficients versus the instantaneous angle of attack are presented in Figure 7 and Figure 8, respectively. These coefficients show the variation as a function of angle of attack during a cycle of motion. The comparisons show that the coefficients obtained by the flow solver are nearly close to the experimental data. This demonstrates the validity of the flow solver and the applicability of the SVR deformation method.

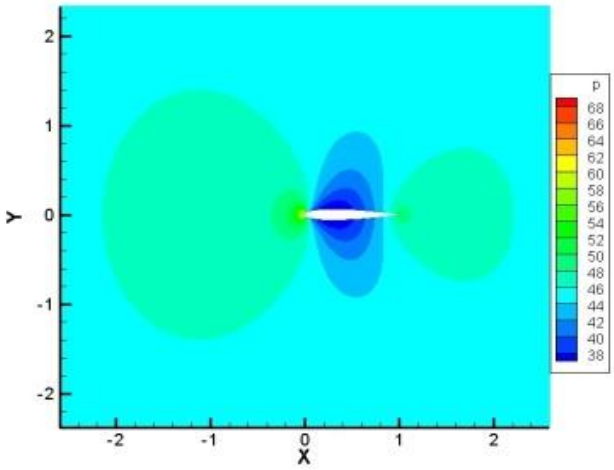

Figure 5. Pressure Field around the Airfoil at Initial 0.016° of Attack

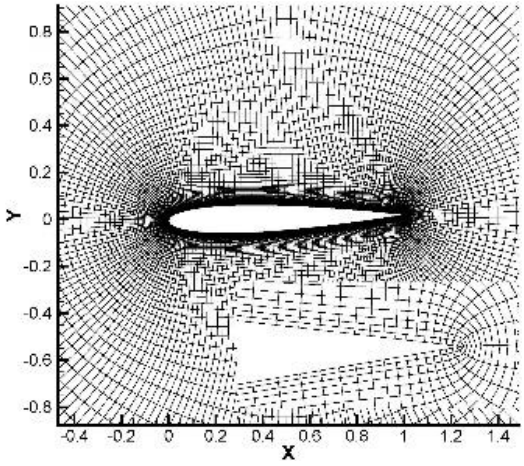

Figure 6. The Deformed NACA0012 Mesh Using the SVR Method

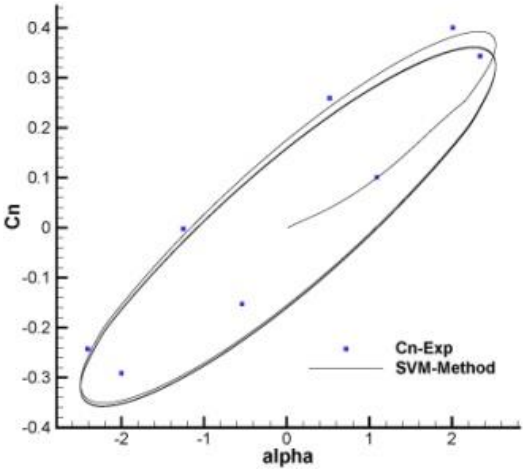

Figure 7. Comparison of the Normal Coefficient with Experimental Data

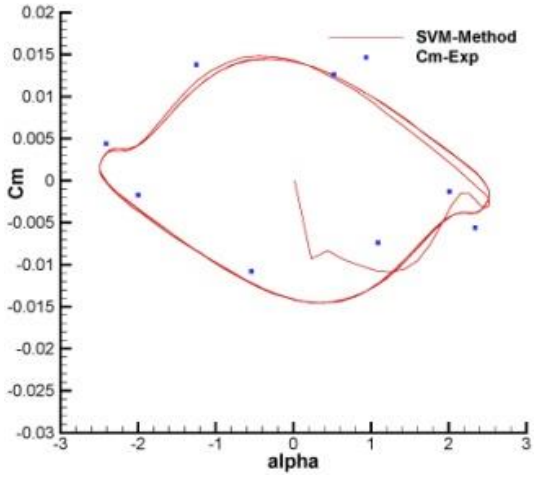

Figure 8. Comparison of the Moment Coefficient with Experimental Data

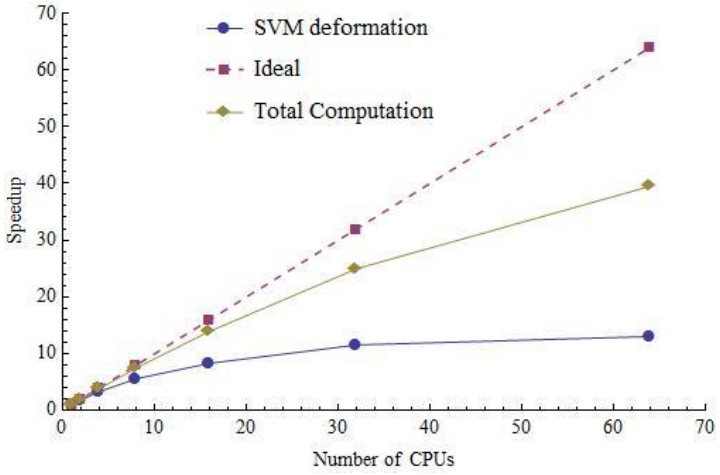

Figure 9. Parallel Speedup of the SVR Method

The parallel speedup of the motion solver relative to the number of CPU cores is presented in Figure 9. Due to the regression function is computed in all cores, the time cost of this part cannot be reduced while the number of CPU cores increases. Therefore, the pure speedup of the "SVR deformation" section is not high. When the number of CPUs is 64, the speedup of this section is only about 13. However, when it coupled with CFD calculation, the parallel efficiency of the total computation is over 60% with 64 CPU cores.

4. Conclusions

In this work, we have developed a parallel mesh deformation method based on support vector machine. This machine learning algorithm solves a simple quadratic programming problem by automatically selecting key boundary points to form the regression function. After the regression function is computed in each CPU, we use it to predict the displacements of internal points in each computational subdomain simultaneously.

The proposed method is tested with the Gaussian kernel function for a typical pitching airfoil case. The results demonstrate that, with proper parametric setting, the deforming capability and efficiency of the SVR method is good enough to perform large-scale numerical simulations. In general, the Gaussian kernel width σ can set close to the length of the deformable part of the object, and the regression threshold ε should be less than the finest mesh spacing divided by the number of deforming steps to avoid mesh cells intersecting.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No. 11502296 and No. 61561146395; the Foundation of National University of Defense Technology under Grant No. ZDYYJCYJ20140101.

References

Li, X., Wang, H., Gu, B., & Ling, C. X. (2015). Data Sparseness in Linear SVR. *International Conference on Artificial Intelligence*. AAAI Press.

Li, K., Yin, J., Lu, Z., Kong, X., Zhang, R., & Liu, W. (2013). Multiclass boosting SVR using different texture features in HEp-2 cell staining pattern classification. *ICPR 2012, Tsukuba* (pp. 170-173).

Platt, J. C. (1998). Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. *Microsoft Research Technical Report*.

Landon, R. H. (1982). NACA0012 oscillation and transient pitching. In *Compendium of Unsteady Aerodynamic Measurements, Advisory Report 702*.

Burges, C. J. C. (1998 January). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, *2*, 121-167. https://doi.org/10.1023/A:1009715923555

Sheng, C., & Allen, C. B. (2003 November). Efficient Mesh Deformation Using Radial Basis Functions on Unstructured Meshes. *AIAA Journal*, *51*, 707-720. https://doi.org/10.2514/1.J052126

Cherkassky, V., & Ma, Y. (2004 January). Practical selection of SVR parameters and noise estimation for SVR regression. *Neural Networks*, *17*, 113-126. https://doi.org/10.1016/S0893-6080(03)00169-2

Zhang, L., & Wang, Z. J. (2004 August). A block LU-SGS implicit dual time-stepping algorithm for hybrid dynamic meshes. *Computers & Fluids*, *33*, 891-916. https://doi.org/10.1016/j.compfluid.2003.10.004

De Boer, A., Van Der Schoot, M. S., & Bijl, H. (2007 June-July). Mesh deformation based on radial basis function interpolation. *Computers & Structures*, *85*, 784-795. https://doi.org/10.1016/j.compstruc.2007.01.013

Pal, S., Basak, D., & Patranabis, D. C. (2007 October). Support Vector Regression. *International Journal of Neural Information Processing Letters and Reviews*, *11*, 203-224.

Rendall, T. C. S., & Allen, C. B. (2009 September). Efficient mesh motion using radial basis functions with data reduction algorithms. *Journal of Computational Physics*, *228*, 6231-6249. https://doi.org/10.1016/j.jcp.2009.05.013

Daude, F., Galon, P., Gao, Z., & Blaud, E. (2014 May). Numerical experiments using a HLLC-type scheme with ALE formulation for compressible two-phase flows five-equation models with phase transition. *Computers & Fluids*, *94*, 112-138. https://doi.org/10.1016/j.compfluid.2014.02.008