SVM-Based Topological Optimization of Tetrahedral Meshes

Xiaoshen Chen, Dewen Peng, Shuming Gao*

(State Key Lab. of CAD&CG at Zhejiang University, Hangzhou, China, 310027)

smgao@cad.zju.edu.cn

Abstract

Finite element analysis （FEA） has been widely used in various fields of industrial product analysis. During the whole process of FEA, mesh model generation plays a key role, which directly influences the accuracy and speed of FEA. In order to generate high quality mesh, a number of topology-based mesh optimization methods have been proposed and applied. However, they are all quite time consuming. In this paper, we propose a SVM-based approach to topological optimization of tetrahedral meshes, aiming to improve the efficiency of topological mesh optimization by using machine learning technique. First, the methodology of the SVM-based topological mesh optimization is put forward. Then the specific features for three kinds of flip operations for tetrahedral meshes are identified and the corresponding SVM models are further set up. Finally three SVM-based flip operations are implemented and the approach is verified and analyzed. The experiment result shows the SVM-based mesh optimization method can improve the mesh optimization efficiency without losing mesh quality.

Keywords: Mesh Optimization, Finite Element Analysis, SVM, Edge Collapse, Flip

Introduction

The application of finite element method (FEM) generally starts with discretizing a continuous CAD (Computer Aided Design) model into tetrahedral meshes by automatic mesh generation tools. However, the meshes directly generated usually do not meet the requirements of qualitative solution computations due to the low quality meshes or distorted meshes generated. Therefore a lot of mesh improvement methods have been proposed to optimize the mesh quality. In general, the current mesh optimization methods can be classified into two categories, smoothing and topological transformation.

Smoothing improves the mesh quality by repositioning the vertices of meshes without modifying its topology. The most famous smoothing algorithm is Laplacian Smoothing proposed by L.R.Hermann[1], which moves each internal vertex of the meshes to the average of its adjacent neighbors. However, it is inevitable to produce tetrahedrons of poor quality. V. N. Parthasarathy and S. Kodiyalam[2] proposed a better smoothing method via defining an objective function and using numerical optimization method to carry out the smoothing process. L. Freitag, M. Jones, and P. Plassman[3] proposed an optimization method whose objective function is not continuous. Consequently, the improvement of the adjacent tetrahedron group of a vertex can take place regardless of the identical change of the worst tetrahedron in this group.

On the other hand, topological transformation improves the mesh quality by adjusting the connectivity of the meshes. It is a local mesh improvement method, where only the meshes within a certain area are changed. The most common topological transformations for tetrahedral meshes are 2-2 flip, 2-3 flip and 3-2 flip[4]. The 2-2 flip deletes the two tetrahedrons on the same coplanar boundary face and replaces them with two new tetrahedrons. The 2-3 flip deletes the two tetrahedrons sharing a face and replaces them with three new tetrahedrons with a new edge shared. The 3-2 flip is just the inverse operation of 2-3 flip, which deletes the three tetrahedrons sharing an edge and replaces them with two tetrahedrons sharing a face. The more generalized topological transformations are edge removal and multi-face removal. Edge removal, proposed by E. Briere de L'Isle and P.L. George[5], replaces m tetrahedrons with 2m-4 new tetrahedrons (2m-2 if the edge is on the mesh boundary), while multi-face removal[6] replaces 2m tetrahedron with m+2 new tetrahedrons. In addition, edge contraction originally used for coarsen[7, 8] is adopted to optimize meshes by B.M. Klingner[4], which removes an edge and deletes all the meshes sharing that edge. B.M. Klingner and J.R. Shewchuk[6] also used vertex insertion to improve a particularly bad tetrahedron of the mesh by inserting a new vertex within its interior or on its boundary.

Support vector machine (SVM)[9] is an effective machine learning method used for classification and regression analysis. SVM usually takes a set of input data for behavior training, through which an SVM model can be generated and used to predict future events. An SVM model is a representation of space division by putting gap line according to the distribution of the input data and its classification. Its goal is to make the gap between each separate class as wide as possible so that the classification is clear. With the availability of the built SVM model, new examples can then be mapped into the same space and predicted according to which side they are to the gap line.

It is observed that topological transformation is very important for improving tetrahedral mesh quality on the one side; on the other side, it is too time-consuming to use for large scale meshes due to its very low success rate. According to B.M. Klingner[4], the success rate of topological operations such as flip operation and edge contraction is below 40%.

In order to improve the efficiency of the topological transformation, the bottle-neck of its practical application, we put forward in this paper a SVM-based approach to topological optimization of tetrahedral meshes, which is intended to improve the efficiency of topological transformation by improving its success rate with the help of machine learning technique.

Overview of the approach

According to the basic procedure of general topological mesh optimization approaches, as shown in Figure 1, each topological operation has to be performed on every tetrahedron of the tetrahedral mesh model, and if it fails (i.e. the topological operation does not improve the tetrahedron's quality), the program has to roll back. Since the amount of the tetrahedrons whose quality need to be improved and can be improved by topological operations are generally very small, most topological operations generally fail. In other words, the time spent on failed topological operations is very big and in essence unnecessary. Obviously, this is one of the major reasons that cause the current topological mesh optimization too time-consuming.

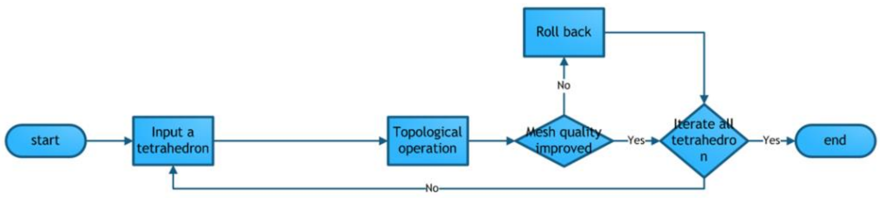

Fig. 1. The procedure of current topological mesh optimization

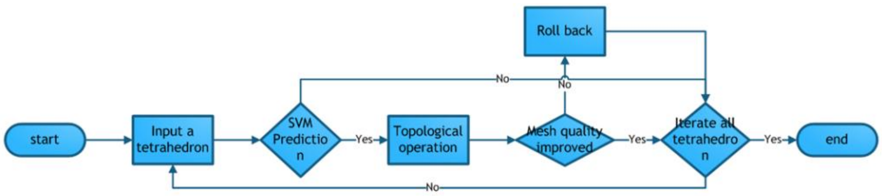

Fig. 2. The procedure of SVM-based topological mesh optimization

Based on the above analysis, in order to improve the efficiency of topological mesh optimization, one possible solution is to limit topological operations as much as possible onto those tetrahedrons whose quality can be indeed improved by the topological operations. And the key point here is how to effectively determine such tetrahedrons. In view that there has been no way to directly and precisely determine such tetrahedrons up to now, we adopt the SVM technique to determine such tetrahedrons through its powerful classification function. Specifically, SVM is used to classify all tetrahedrons into two categories with respect to the topological operation: the tetrahedrons that should accept the topological operation according to the sample learning, and the tetrahedrons that should not. With the help of SVM-based classification, the topological operation can then be performed only on those tetrahedrons that the SVM model predicts they should accept the topological operation. This way, a large number of unnecessary topological operations along with the corresponding roll-back operations can be avoided during the topological mesh optimization process, and thus considerable time can be saved. Figure 2 shows the procedure of our SVM-based topological mesh optimization approach.

The proposed approach consists of two parts: the construction of the SVM models for topological operations and the features selection involved in the SVM model construction, and details of them are described below.

Construction of the SVM models

To achieve the SVM-based classification on tetrahedrons with respect to the topological operation, a proper SVM model should be constructed first. Generally the construction of the SVM model consists of the following six steps. Note that, in this work, we use libSVM as a tool to implement SVM classification, which is an integrated software for support vector classification, regression and distribution estimation developed by Chih-Chung Chang and Chih-Jen Lin[10].

1. Select the appropriate features and form the feature space to characterize the classification. In view that this step is critical and related to the specific topological operation, we will describe it in details in the next section.
2. Construct training set according to the selected features. We imitate the method of cross-validation to construct training set, which randomly picks up a certain percentage of data as the training set, while others are used as testing data. After have the training set, we extract the feature information from the training set and generate a training file according to the feature order. The training file set up is for sample learning.

3. Scale the training set. Specifically, scale all features in an appropriate range, mostly in [0, 1]. We do this with the svm-scale function provided by libSVM.

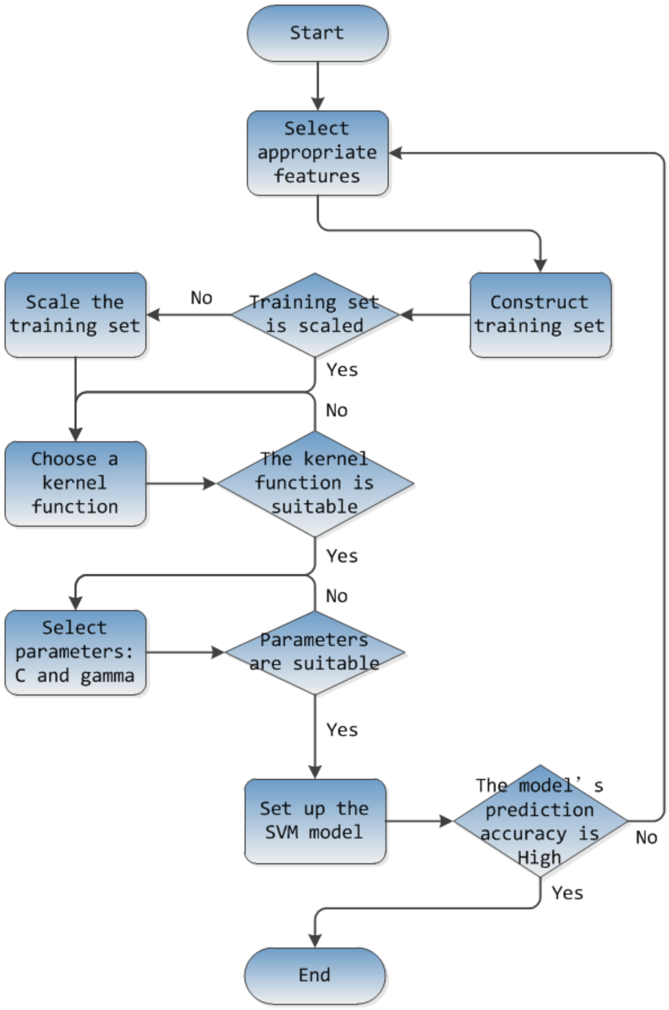

Fig. 3. The flow chart of the SVM model construction

4. Choose a suitable kernel function. To find a proper kernel function that can better separate the feature space, all kernel functions are tested with all the training samples, and the radial basis function is found to have the highest prediction accuracy averagely, and it is thus chosen as our kernel function.
5. Searching for the best parameters C and gamma. C is the penalty co-

efficient used to define how much error can be tolerated, and gamma is related to the kernel function being chosen, influencing the final space distribution. These two parameters much influence the SVM prediction accuracy and have to be carefully selected. In this work, the parameter selection is conducted using the parameter searching tool provided by libSVM, which uses a brutal-force approach to find the best parameters.

6. Set up the SVM model and test the prediction accuracy. Based on the above steps, a SVM model can then be set up with libSVM. After this, its prediction accuracy is tested. If the prediction accuracy is low, go back to step 1 to modify the features. Such process is iterated until the required prediction accuracy is reached.

Selection of the features for SVM classification

The SVM model is set up through sample learning and the effect of sample learning strongly relies on the selected features used to characterize the classification. Therefore, in order to guarantee the accuracy of the SVM classification for each type of topological operation, the proper features of the topological operation should be selected. Our criterion for selecting features is that the selected features should be able to reflect the difference between the tetrahedrons whose quality can be improved by the topological operation and those whose quality cannot be improved by the topological operation.

The selection of the proper features involves mesh quality measure. In this work, we choose minimum sine[6], volume-length[11] and Jacobian determinant[12] as tetrahedron quality measures. Minimum sine of a tetrahedron's six dihedral angles penalizes both small and large dihedral angles. Volume-length denoted by V / ℓ_{rms}^3 is the signed volume of a tetrahedron divided by the cube of its root-mean-squared edge length. Usually it multiplies $6\sqrt{2}$ to get normalized. Jacobian determinant is the determinant of the Jacobian matrix of a tetrahedron's four vertexes. Usually it is used in normalized form and multiplies a transformation matrix.

For different kinds of topological operations, different features are usually needed. In this work, the topological operations considered include three most commonly used flip operations: 2-2 flip, 2-3 flip and 3-2 flip. Below we describe the features selected for 2-2 flip, 2-3 flip and 3-2 flip respectively.

The features for 2-2 flip

Figure 4 illustrates a 2-2 flip which deletes the internal face shared by two adjacent tetrahedrons and replaces it with a crossing face. Here the boundary faces of the two tetrahedrons are coplanar. In view that the effect of

2-2 flip depends on the specific mesh quality measure, we choose different features for the 2-2 flip with different mesh quality measures.

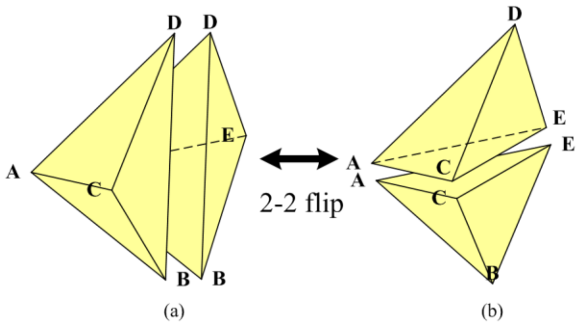

Fig. 4. Illustration of 2-2 flip

For the 2-2 flip with volume-length measure, we choose the following features:

1. Volume-length of tetrahedron ABCD. It is used to indicate the original quality of a tetrahedron ABCD, showing whether there is room for improvement. Meanwhile, volume-length also reflects whether the tetrahedron is a “fat” mesh or a spire mesh.
2. Volume-length of tetrahedron ABDE. It is selected for the same reason as the volume-length of tetrahedron ABCD.
3. ℓ_{BD} and ℓ_{CE} length ratio. When the length of boundary edge BD is longer than that of CE, it is likely that the two new tetrahedrons resulted from 2-2 flip become fatter than un-flipped, which usually means better quality. Otherwise, the resulted tetrahedrons may be more spire than un-flipped. Therefore such length ratio of two tetrahedrons indicates whether fatter tetrahedrons can be obtained through 2-2 flip.
4. Area-length of face BCD. Area-length denoted by $Area / \ell_{rms}^2$ indicates whether the face is a thin triangle or a fat one. Meanwhile the comparison between the area-length of BCD and that of BDE shows whether one face is much bigger than the other or not, and if so, better quality tetrahedrons can be obtained through 2-2 flip. Because of these functions, area-length of BCD is selected as a feature.
5. Area-length of face BDE. It is selected for the same reason as the area-length of face BCD.

For the 2-2 flip with minimum sine measure, we add the following two types of features:

1. Three dihedral angles around vertex C. Dihedral angles are intro-

duced as the feature related to minimum sine. Meanwhile, these three dihedral angles characterize the shape of the tetrahedron around vertex C.
2. Three dihedral angles around vertex E. They are selected as features with the same reason described above.

For the 2-2 flip with Jacobian determinant measure, we remove the first and the second features from the features selected for the 2-2 flip with volume-length measure, and meanwhile add the following two types of features for it:

1. Jacobian determinants of vertex C and vertex E. The Jacobian matrix of a vertex characterizes the shape of the three edges adjacent to the vertex. Meanwhile, the Jacobian determinants of vertex C and vertex E reflect the proportion between the two opposite angles at vertex C and vertex E.
2. Jacobian determinant of tetrahedron ABCD and tetrahedron ABDE. The Jacobian determinant of a tetrahedron refers to the one with worst value among the four Jacobian determinants of the tetrahedron's four vertices, which characterizes the quality of a tetrahedron.

The features for 2-3 flip

2-3 flip is illustrated in Figure 6, which deletes the shared face of two adjacent tetrahedrons and meanwhile creates a new edge to connect two opposite vertices. Similar to 2-2 flip, different features are chosen for the 2-3 flip with different mesh quality measures.

For the 2-3 flip with volume-length measure, we choose the following features:

1. Volume-length of tetrahedron ABCD. It is selected as a feature with the same consideration as that for 2-2 flip.
2. Volume-length of tetrahedron ABCE. It is selected for the same reason as that of the volume-length of tetrahedron ABCD.
3. $\ell_{DE}^2 / Area_{ABC}$. This is selected as a feature because it well characterizes the change brought about by 2-3 flip. In addition, it indicates whether two tetrahedrons are long and narrow with respect to the shared face between them. Specifically, when its value is big, two tetrahedrons are long and narrow; otherwise they are fat.

For the 2-3 flip with minimum sine measure, we add the following features:

1. Minimum sine of tetrahedron ABCD and tetrahedron ABCE. Minimum sine shows the quality of a tetrahedron.

For the 2-3 flip with Jacobian determinant measure, we remove the first and the second features from the features selected for the 2-3 flip with volume-length measure, and meanwhile add the following two types of features into it:

1. Jacobian determinants of vertex D and vertex E.
2. Jacobian determinant of tetrahedron ABCD and tetrahedron ABCE. They indicate the quality of the original tetrahedrons and also show whether there are room for improvement.

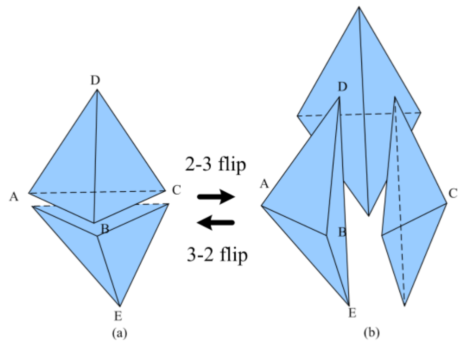

Fig. 6. Illustration of 2-3 flip and 3-2 flip

The features for 3-2 flip

3-2 flip is the opposite operation to 2-3 flip, as illustrated by Figure 6. For 3-2 flip, we select the following features no matter which mesh quality measure is used:

1. Volume-length of tetrahedron ABDE, tetrahedron BCDE and tetrahedron ACDE. They are selected as features with the same consideration as that for 2-2 flip and 2-3 flip.
2. Area-length of face ADE, face BDE and face CDE. Each of them indicates whether the corresponding face is a thin triangle or a fat one. Meanwhile the comparison between them indicates whether they are similar or not.

RESULTS AND DISCUSSIONS

The proposed approach to SVM-based topological optimization of tetrahedral meshes has been implemented, using libSVM as a tool to implement SVM classification. The program is running on a PC with a Core2 Q9400 2.66 GHz CPU and 4G Memory. The algorithm has been tested by 17 industrial models shown in Figure 7. Tables 1-3 summarize~~s~~ the test results. In these tables, for each test model, there are three lines of data and each line corresponds to a kind of mesh quality measure, i.e. Jacobian determinant, or minimum sine, or volume-length respectively. The original time in the tables refers to the total time of performing the traditional flip operations, and the SVM-based time in the tables is the total time of performing the SVM-based flip operations, including the time spent in predicting. Note that N/A in Table 3 means there is no any edge in the model shared by three tetrahedrons, which is the prerequisite of performing 3-2 flip.

Table 1. Flip2-2 test results

Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)	Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)
1	58037	91.46%	3.99	3.85	e1	20079	88.38%	0.68	0.65
		97.22%	3.59	3.67			94.15%	0.65	0.64
		96.27%	3.77	3.57			92.72%	0.67	0.6
cad_cube _out	362	100%	0.011	0.008	gear10k	10058	76.28%	0.28	0.27
		92.31%	0.011	0.008			79.62%	0.27	0.27
		92.31%	0.012	0.008			80.64%	0.27	0.26
Clg	26482	90.59%	1.01	0.93	house2	1389	84.90%	0.035	0.033
		95.16%	0.97	0.91			88.86%	0.034	0.032
		94.60%	1	0.86			90.35%	0.033	0.03
Cow	42053	99.06%	1.75	1.12	P	926	83.68%	0.024	0.023
		98.99%	1.82	1.41			88.89%	0.021	0.022
		99.17%	1.93	1.08			87.15%	0.021	0.02
crank_arm	13157	86.98%	0.39	0.35	rand1	5104	87.70%	0.103	0.102
		94.97%	0.4	0.35			84.76%	0.102	0.102
		93.43%	0.39	0.33			88.21%	0.105	0.099
cube1k	1184	97.88%	0.042	0.027	rand2	25704	86.94%	0.727	0.619
		97.89%	0.041	0.026			83.31%	0.658	0.583
		99.30%	0.045	0.028			88.33%	0.685	0.571
cube10k	11660	99.90%	0.43	0.25	Tfire	1104	89.41%	0.032	0.018
		99.70%	0.43	0.25			95.29%	0.032	0.018
		100%	0.43	0.24			90%	0.033	0.017
Dragon	32959	99.49%	1.32	0.926	Tire	11098	91.78%	0.334	0.139
		99.26%	1.34	1.2			96.16%	0.329	0.137
		99.65%	1.37	0.84			92.64%	0.333	0.133
e0	46471	91.60%	2.42	2.19					
		97.06%	2.57	2.15					
		96.35%	2.46	2.01					

Table 2. Flip2-3 test results

Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)	Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)
1	58037	99.97%	2.314	1.304	e1	20079	100%	0.731	0.516
		99.97%	2.17	1.18			100%	0.555	0.433
		99.68%	2.151	1.373			100%	0.548	0.522
cad_cube_out	362	100%	0.0097	0.0025	gear10k	10058	100%	0.322	0.277
		100%	0.0098	0.0025			100%	0.259	0.24
		100%	0.0122	0.003			100%	0.259	0.255
Clg	26482	99.97%	0.968	0.657	house2	1389	100%	0.034	0.032
		99.97%	0.782	0.563			100%	0.036	0.029
		99.97%	0.751	0.667			100%	0.034	0.034
Cow	42053	99.99%	2.084	1.079	P	926	100%	0.021	0.019
		100%	1.704	0.917			100%	0.021	0.017
		99.96%	1.692	1.064			100%	0.023	0.022
crank_arm	13157	100%	0.395	0.33	rand1	5104	86.63%	0.118	0.076
		100%	0.339	0.289			87.18%	0.116	0.071
		100%	0.341	0.335			82.32%	0.109	0.082
cube1k	1184	100%	0.042	0.011	rand2	25704	83.86%	0.879	0.43
		100%	0.043	0.011			84.32%	0.677	0.351
		100%	0.043	0.011			80.15%	0.639	0.436
cube10k	11660	100%	0.506	0.428	Tfire	1104	99.79%	0.032	0.029
		100%	0.431	0.378			100%	0.035	0.025
		99.99%	0.435	0.371			99.69%	0.032	0.032
Dragon	32959	99.98%	1.572	0.713	Tire	11098	99.93%	0.429	0.325
		99.99%	1.25	0.593			99.98%	0.34	0.311
		99.92%	1.241	0.676			99.88%	0.336	0.335
e0	46471	99.97%	2.066	1.155					
		99.98%	1.635	0.974					
		99.97%	1.644	1.147					

Table 3. Flip3-2 test results

Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)	Models	Num of Tetras	Prediction accuracy	Original time (s)	Svm-based time (s)
1	58037	98.26%	2.016	2.027	e1	20079	98.18%	0.518	0.518
		99.02%	1.967	1.947			99.09%	0.517	0.51
		98.08%	1.873	1.82			97.92%	0.446	0.445
cad_cube _out	362	100%	0.0049	0.0048	gear10k	10058	100%	0.221	0.212
		100%	0.0048	0.0046			100%	0.219	0.21
		100%	0.0038	0.0037			99.88%	0.193	0.186
Clg	26482	98.48%	0.771	0.774	house2	1389	100%	0.023	0.022
		99.39%	0.769	0.761			100%	0.024	0.023
		98.65%	0.693	0.692			99.31%	0.02	0.019
Cow	42053	N/A	N/A	N/A	P	926	100%	0.015	0.014
		N/A	N/A	N/A			100%	0.015	0.015
		N/A	N/A	N/A			100%	0.013	0.012
crank_arm	13157	98.47%	0.323	0.324	rand1	5104	61.38%	0.11	0.107
		99.49%	0.321	0.318			54.50%	0.112	0.109
		98.39%	0.289	0.289			64.02%	0.092	0.09
cube1k	1184	N/A	N/A	N/A	rand2	25704	56.94%	0.79	0.784
		N/A	N/A	N/A			54.74%	0.802	0.8
		N/A	N/A	N/A			62.04%	0.701	0.701
cube10k	11660	N/A	N/A	N/A	Tfire	1104	N/A	N/A	N/A
		N/A	N/A	N/A			N/A	N/A	N/A
		N/A	N/A	N/A			N/A	N/A	N/A
Dragon	32959	N/A	N/A	N/A	Tire	11098	91.30%	0.167	0.16
		N/A	N/A	N/A			73.91%	0.169	0.163
		N/A	N/A	N/A			86.96%	0.17	0.163
e0	46471	97.03%	1.413	1.447					
		98.97%	1.415	1.412					
		97.65%	1.266	1.287					

From Table 1-3, we can see that, for the SVM-based 2-2 flip and 2-3 flip, the average prediction accuracy is above 90% and the average percentage of the time saved compared with the traditional 2-2 flip and 2-3 flip is about 10%. However, for the SVM-based 3-2 flip, its prediction accuracy is low for certain models like rand1 and rand2 in Figure 7; and its efficiency is not as good as that of 2-2 flip or 2-3 flip and even could be slower than that of the traditional 3-2 flip for few models such as e0 in Figure 7. By analyzing the reason for this, we find that the high success rate of 3-2 flip is the key factor. According to our test, the success rate of 3-2 flip is averagely above 80%. It means that the most original 3-2 flip operations can improve the quality of the related tetrahedrons, and thus it is not necessary to do additional prediction by SVM for them. From Table 1-3, we can also find that, for each kind of flip operation, the difference between prediction rate and the running time caused by taking different mesh quality measures is minor, averagely below 5%.

The experimental results show that the SVM-based method can effectively improve the efficiency of topological mesh optimization. And obviously the more sophisticated a topological operation is, the greater the effect of its corresponding SVM-based operation will be.

CONCLUSIONS

This paper presents a novel SVM-based approach to topological optimization of tetrahedral meshes. The approach exploits SVM to predict whether the quality of a tetrahedron can be improved by performing the topological operation on it first, and then determines whether to conduct the topological operation on the tetrahedron according to the prediction result. In this way, the time-consuming topological operations can be limited as much as possible, only onto those tetrahedrons whose quality can be really improved by the topological operations. Consequently, considerable time can be saved by avoiding large number of unnecessary topological operations along with the corresponding roll-back operations. As the first step of the research, three SVM-based flip operations are implemented, and the experiment results show the potential of the SVM-based mesh optimization approach.

Our future work mainly consists of three aspects. First, construct SVM models for more sophisticated topological operations. We have already begun to construct SVM model for edge contraction, which consumes more running time and usually has less than 10% success rate. Second, construct SVM models for composite mesh improvement methods. Since topological operations are usually performed together with smoothing afterwards, a better SVM model should be able to predict these composite operations together. Third, extend the proposed approach to topology-based hexahedron mesh optimization.

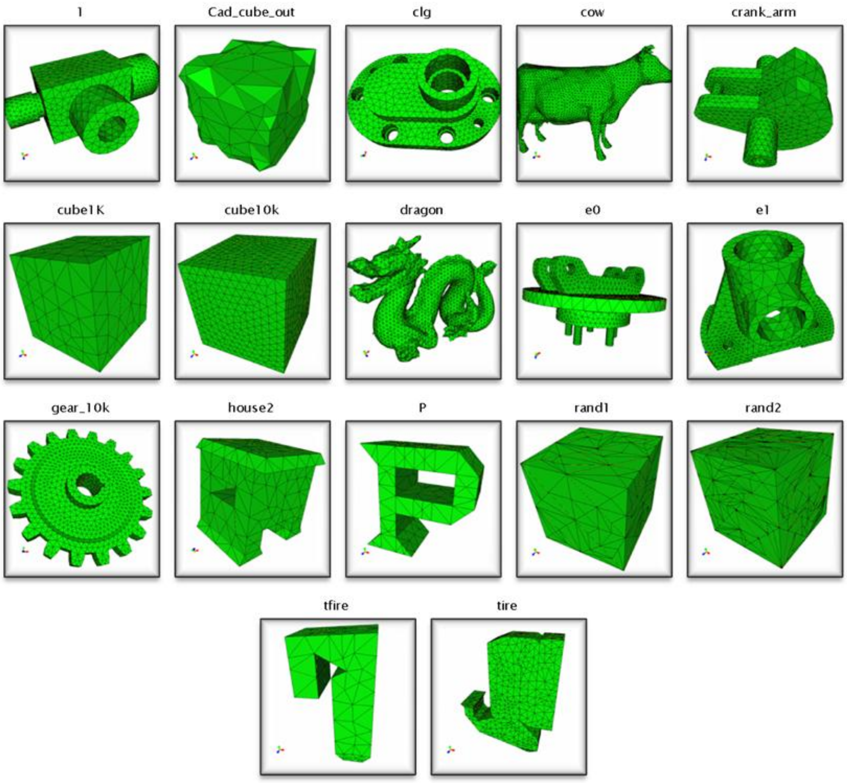

Fig. 7. Test models

ACKNOWLEDGMENT

The authors are very grateful to the financial support from NSF of China (No.60736019).

REFERENCES

[1]. Herrmann, L.R., Laplacian-isoparametric grid generation scheme. Journal of the Engineering Mechanics Division, 1976. 102(5): p. 749--907.

[2]. Parthasarathy, V.N. and S. Kodiyalam, A constrained optimization approach to finite element mesh smoothing. Finite Elements in Analysis and Design, 1991. 9(4): p. 309--320.

[3]. Freitag, L., M. Jones and P. Plassmann. An E cient Parallel Algorithm for

Mesh Smoothing. 1995.
[4]. Klingner, B.M., Improving tetrahedral meshes. 2008: ProQuest.
[5]. Briere de L'Isle, E. and P.L. George, Optimization of tetrahedral meshes. Institute for Mathematics and Its Applications, 1995. 75: p. 97.
[6]. Klingner, B.M. and J.R. Shewchuk. Aggressive tetrahedral mesh improvement. 2008.
[7]. Ollivier-Gooch, C., Coarsening unstructured meshes by edge contraction. International journal for numerical methods in engineering, 2003. 57(3): p. 391--414.
[8]. Dey, T.K., et al., Topology preserving edge contraction. Publ. Inst. Math. (Beograd)(NS), 1999. 66(80): p. 23--45.
[9]. Cortes, C. and V. Vapnik, Support-vector networks. Machine learning, 1995. 20(3): p. 273--297.
[10]. Chang, C.C. and C.J. Lin, LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST), 2011. 2(3): p. 27.
[11]. Parthasarathy, V.N., C.M. Graichen and A.F. Hathaway, A comparison of tetrahedron quality measures. Finite Elements in Analysis and Design, 1994. 15(3): p. 255--261.
[12]. Freitag, L.A. and P.M. Knupp, Tetrahedral mesh improvement via optimization of the element condition number. International Journal for Numerical Methods in Engineering, 2002. 53(6): p. 1377--1391.