Titre: Title:	Automatic hybrid grid generation
Auteur: Author:	Yuan Li Wang
Date:	2007
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Wang, Y. L. (2007). Automatic hybrid grid generation [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8066/

UNIVERSITÉ DE MONTRÉAL

AUTOMATIC HYBRID GRID GENERATION

YUAN LI WANG
DÉPARTEMENT DE GÉNIE MÉCANIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE MÉCANIQUE)
AUGUST 2007

Your file Votre référence
ISBN: 978-0-494-35519-0
Our file Notre référence
ISBN: 978-0-494-35519-0

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

AUTOMATIC HYBRID GRID GENERATION

présentée par : WANG YUAN LI

en vue de l'obtention du diplôme de : Philosophiæ Doctor

a été présentée au jury d'examen constitué de :

Mr. TRÉPANIER Jean-Yves, Ph.D., président

M. CAMARERO Ricardo, Ph.D., membre et directeur de recherche

M. GUIBAULT François, Ph.D., membre et codirecteur de recherche

Mrs. FARINAS Marie-Isabelle, Ph.D., membre

Mr. CUILLIÈRE Jean-Christophe, Ph.D., membre

To all my family members.

Without your support, this would not have been possible.

ACKNOWLEDGMENTS

I would like to thank my friend, Bin Chen, who opened the door for my PhD research in Ecole Polytechnique de Montreal, a French speaking university. It was in the summer of 2001, Bin introduced me to know Francois, a professor in Computer Science Department. Although my French was very poor at that time, Francois still offered me an opportunity to join his research group. And through Francois, I was introduced to Ricardo, a professor in Mechanical Department. Later Ricardo became my other adviser to guide my research.

I would like to express my sincere gratitude to my advisers. I acquired a lot of confidence through their gentle encouragements. They have helped me to build a solid background knowledge of mesh generation in CFD and expanded my vision in this field. They have taught me the way of how to find out the critical problem during research especially when facing complicated situations.

Their treasure supports from both professional knowledge and financial aspects are so important to me during these four and half years research. For quite a long time, they have fixed almost the whole morning every week for me to discuss my research issues. They have also spent a lot of time in correcting my proposals, papers, and thesis.

I also would like to take this chance to thank the following people who have ever contributed to my research:

Ko-Foa has spent many hours in discussing with me, and given me many insightful advices in mesh generation, he also supplied me 2D segment-segment and 3D segment-triangle intersection detection algorithms.

Ying has taught me a lot in using softwares in the lab and producing many models and test cases

for me to test my algorithms. She speaks good French. She is like my eyes and ears, because she has to translate people's talking to me during the meetings.

Marie-Gabrielle and Julien have also spent a lot of time with me in discussing the unstructured mesh generation, mesh optimization and mesh quality measurement algorithm, they also have given me many technical advises and opinions to improve my presentations. Julien has also taught me about how to produce high quality slides. He is also an experts in using VU, a visualization software used in my lab. I also benefit a lot from him about advanced techniques of using Latex to produce vivid presentations, and posters.

I owe many thanks to the peoples who work in my lab: Man, Sébastien, Jean-Francois, Horea, Xavier, Melisa, Ellisabelle, Charles and Yun. They have provided me a friendly and cooperative research environment. Mohammad and Noa, though they have left the lab long time ago, have also given me many helps in producing test cases and implementation issues.

Many special thanks to the committee members: Dr. Jean-Christophe Cuillière, Dr. Jean-Yves Trépanier and Dr. Marie-Isabelle Farinas for their invaluable support, time and comments.

Finally, I would like to express my appreciation to GE Canada (Hydro) and the National Science and Engineering Research Council of Canada (NSERC) for their support. Thank you, Mr. Thi Vu from GE Canada for supplying many real industrial test cases for me.

ABSTRACT

One of the critical issues addressed very often both in research and industry is the boundary mesh generation technique for wall dominated phenomena, for example, viscous fluid flow, heat transfer, etc. This technique is called front propagation or front offset. This thesis presents a novel application of front propagation technique to domain partitioning and mesh generation processes.

The propagation process proposed in this work is directly inspired from marching technology, that is all the points located on the original front are propagated along their local normal directions. The main difference between the current method and traditional marching methods lies in the way that local normal direction is computed. Traditionally, the local normal directions are computed using geometric information, such as the average normal of neighboring points or facets surrounding the point to be propagated. In this method, the local normal directions are calculated using equation $\vec{n} = \nabla\phi/|\nabla\phi|$. Function ϕ is the numerical solution of the minimum distance equation, which is a variation of the Eikonal equation, $\nabla\phi \cdot \nabla\phi = 1$.

The benefit of calculating normal directions in such a way is that self-intersections are avoided in a natural way. Since normal directions are represented using the numerical solution of the PDE, propagation is thus performed in the ϕ space rather than geometric space. In addition, the proposed method transports the original parameterization to the propagated surface, there is a one-to-one parametric consistency between the original front and its offset, which allows rigorous matching of block interfaces.

This front propagation method used was validated from two aspects: accuracy and efficiency. For accuracy, the results show that the proposed technique converges linearly with mesh size. The result follows from the fact the ϕ field is solved using a first order numerical scheme. The time

spent in this propagation method is divided into three parts, and measured separately: *initialization time* and *fast sweeping time*, which both are dependent on the mesh size; *propagation time*, which is independent of mesh size, but dependent on the initial discretization of the front to be propagated and time increment.

The proposed front propagation technique is successfully applied in the applications of *Geometric Modeling*, ie, offset surface construction. The presented method can be used to offset both facet and NURBS represented geometries. When surfaces are discretized into triangular or quadrilateral surfaces, the discretization points are propagated directly and the offset points are used to construct new offset boundary; when surfaces are NURBS represented geometries, the control points are propagated using the proposed method.

Another type of application presented in this thesis is *Mesh Generation*, ie, boundary mesh generation. The proposed method is used to decompose the entire domain into sub-domains near boundary areas, and then hybrid meshes are generated in each sub-domain. The boundary mesh is validated using a ball valve model under both steady and unsteady flow conditions. The preliminary result shows that the presented method provides a possibility to generate boundary mesh in a robust and automated manner.

RÉSUMÉ

Cette dissertation présente une recherche sur nouvelle technique de propagation de front avec comme finalité son utilisation dans la méthodologie de génération de maillages hybrides dans des configurations industrielles. Le domaine est la simulation numérique d'écoulements complexes dans des applications industrielles telles que les turbomachines. La qualité et l'efficacité de ces solutions dans un processus de design dépend largement du maillage qui sert de support à la discrétisation. Outre la conformité du maillage à la géométrie, la problématique d'actualité se trouve au niveau de l'adaptation à une phénoménologie de l'écoulement, c'est-à-dire on recherche un maillage qui épouse les variations des propriétés de l'écoulement, telles que les champs de vitesses, pression etc. Le défi dans un contexte industriel est de concilier plusieurs exigences contradictoires, la finesse du maillage et le coût de la solution. En effet, un écoulement en dimension trois à l'intérieur des divers éléments d'une turbine présente une grande disparité d'échelle dans ces variables, de sorte qu'un «bon» maillage dans une partie de l'écoulement ne le sera pas ailleurs, ou bien engendrera un coût de calcul trop élevé.

La stratégie proposée dans ce travail repose sur la génération de maillages hybrides qui consiste à utiliser différents types de maillages selon la région du domaine de calcul. Cette approche est bien adaptée aux problèmes basés sur des écoulements fluides où le domaine se divise en une partie fortement rotationnelle où dominent les effets visqueux fortement concentrés dans le voisinage des parois ou bien des trainées des obstacles, et une partie plus homogène à l'extérieur de cette mince couche. Spécifiquement, dans ce travail on entend par maillage hybride des éléments non-structurés isotropes (tétrahèdres) à l'extérieur et des éléments semi-structurés anisotropes (prismes ou hexahèdres) dans les couches près des parois.

La mise en oeuvre de la génération de maillages hybride nécessite donc un découpage ou partition du domaine en zones ou couches selon ce critère de proximité à la paroi. Nous avons étudié les diverses techniques de décomposition de domaines aux vues de leur applicabilité au problème présent. Ces techniques ont comme objectif de découper un domaine quelconque pour le remplacer par plusieurs sous-domaines plus simples géométriquement, c-à-d s'approchant d'un rectangle. À l'origine la plupart des solutions proposées visaient la génération d'un maillage structuré à l'intérieur de chacun des sous-domaines. La caractéristique (et la faiblesse) des partitions ainsi obtenues est un découpage topologique global qui ne «respecte» pas nécessairement bien la géométrie dans le voisinage des parois. Or, dans les configurations envisagées dans cette étude, les sous-domaines que l'on cherche à obtenir doivent être très minces, de la taille d'un couche limite ou d'un sillage, et doivent épouser la paroi. Il en découle certaines exigences particulières notament la nécessité d'engendrer des mailles étirées avec un rapport d'alongement très élevé. On parle effectivement de couches ou de peaux.

Les premières tentatives dans ce sens ont été obtenues par des opérations géométriques qui déplacent («offset» en anglais) la représentation de la géométrie. Une analyse critique montre plusieurs difficultés: 1) lourdeur sur le plan des opérations mathématiques, 2) complexité dans leur utilisation dans un contexte d'ingénierie et 3) certaines pathologies dans les résultats. Sous cette dernière rubrique, nous soulignons le problème de croissement de surfaces. Ces discussions sont présentées au Chapitre 2, mais la plus contraignante est celle du manque de fiabilité des résultats dans une utilisation en mode d'exploitation, où on tente de minimiser les interventions humaines. Pour réaliser, même partiellement cet objectif, on doit baser la technique de déplacement des surfaces sur des algorithmes fiables qui évitent les intersections de façon intrinsèque plutôt que sur des heuristiques comme les méthodes courantes de construction par déplacement géométrique.

Une recherche blibliographique dans le domaine de l'avance de fronts a permis d'identifier des

approches qui semblent rencontrer les diverses exigences du problème de génération de peaux et qui évitent les difficultés associées aux techniques de construction géométriques. La différence principale réside dans le calcul des normales. Dans le cas de ces ces dernières, les directions normales au front sont calculées par la moyennes des vecteurs perpendiculaires à partir de données géométriques, ce qui donne lieu à des croisements du front. Dans la méthode proposée, le problème est reformulé sous la forme d'une équation différentielle, l'équation Eikonale, $\nabla\phi \cdot \nabla\phi = 1$ où ϕ représente une fonction de distance minimale que l'on solutionne sous une forme faible.

La normale est obtenue calculée comme le gradient deϕ, $\vec{n} = \nabla\phi/|\nabla\phi$, L'avantage de calculer les normales de cette façon est que les croisements sont automatiquement et naturellement évités par la nature même de la solution faible de l'équation Eikonale. De plus, avec cette méthode on transporte la paramétisation du front original, ce qui assure une consistance entre les paramétrisations du front transporté et le front initial.

Une implémentation de cette technique de propagation et son application à différents cas industriels constituent les principales contributions de cette thèse.

CONDENSÉ EN FRANÇAIS

La qualité des simulations numériques dépend fortement des diverses techniques de discrétisation appliquées à la géométrie du domaine ainsi qu'aux équations. Spécifiquement en ce qui a trait à la discrétisation spatiale, la précision et l'efficacté des méthodes numériques reposent sur la forme et la répartition des éléments générés par le processus de génération de maillage.

Afin d'améliorer le processus de génération de maillage pour des géométries complexes, on développe dans cette recherche des méthodes nouvelles de décomposition de domaine et de propagation de front, qui peuvent être utilisées dans une stratégie de génération de maillages hybrides. Les méthodologies courantes de maillages hybrides sont étendues avec les objectifs spécifiques suivants:

Étendre les capacités de résolution des caractéristiques physique :

La capacité de simulation de gradients élevés dans des couches limites d'écoulements visqueux dépend de la capacité du maillage à localiser ces régions et à fournir un type et une densité de maille appropriés. Cette recherche vise le développement d'une nouvelle technique de génération de maillage afin de surmonter les faiblesses principales retrouvées dans les approches courantes.

Accroître le niveau d'automatisation :

Bien que de nombreuses techniques de maillages hybrides aient été proposées, le maillage de domaines de complexité arbitraire demeure un frein significatif dans ce processus. Un objectif important de cette recherche est de tenter de développer un algorithme avec un haut niveau d'automatisation qui minimise les interventions usager.

Améliorer l'efficacité du calcul :

L'utilisation de la méthode de balayage rapide (*Fast Sweeping Method*), avec son temps de calcul $\Omega(n)$, où n représente le nombre total de noeuds de maillage du domaine de calcul, se compare avantageusement, en terme de coût de calcul, à d'autres méthodes telles que le *Fast Marching*. Des algorithmes de parallèlisation peuvent également être appliqués à ce schéma de calcul.

La méthode développée dans cette recherche permet d'aborder la simulation d'écoulements présentant une anisotropie élevée telle que rencontrée dans des phénomènes où dominent les effets de paroi comme dans les couches limites dans les applications en aérodynamique. Afin de cerner la directionalité du phénomène, des maillages hautement anisotropes sont nécessaires dans le voisinage des parois. Loin des frontières, des éléments isotropes peuvent être utilisés. D'où le besoin de partitionner le domaine en une couche pariétale ou peau anisotrope, et un coeur isotrope pour le reste du domaine. La région de la couche est générée par une technique de propagation de front qui apporte plusieurs avantages importants par rapport aux méthodes traditionnelles:

1. Une technique de décomposition nouvelle:

   Dans cette technique le problème de décomposition de domaine est formulé comme un problème de déplacement de surface et sa solution est obtenue en utilisant les équations Eikonale. Cette technique peut être appliquée à la solution de problèmes où dominent les parois, par exemple les écoulements visqueux, de transfer de chaleur, etc.

2. Un modèle mathématique simplifié:

   Le nouveau modèle proposé pour résoudre le problème de propagation de front, *l'équation de distance de déplacement* (Offset Distance Equation), est une variante de l'équation

Eikonale. Une validation de l'extension du *schéma de balayage rapide* en 3d est réalisé avec des surfaces lisses. Cette partie du travail est resumée dans la publication Wang *et al.* (to be published), soumis au International Journal for Numerical Methods in Engineering.

3. Un calcul amélioré de la direction normale:

   Les directions normales sont representées en utilisant la solution numérique de l'équation Eikonale, qui évite les intersections de surfaces en propre pendant leur propagation. Les détails sur le contourmenent de ces intersections est discuté au Chapitre 5.

4. Une nouvelle méthode de construction de surface par déplacement:

   Comme une extension, cette approche a été appliquée à la propagation de fronts à géométries complexes, c-à-d des frontières convexes et concaves dans des domaines non-simplement connexes. Spécifiquement, différentes techniques ont été développées pour la construction de telles surfaces de déplacement dans des topologies diverses comme des surfaces perméables pour les conditions frontières d'entrée/sortie.

La méthode proposée a été appliquée avec succès à différentes situations, par exemple, la construction de surfaces par déplacement, le maillage de frontières et l'analyse d'écoulements fluides. Ces travaux sont publiés dans Wang *et al.* (June 2005), Wang *et al.* (September 2005) et dans Wang *et al.* (September 2006) respectivement.

La méthode de déplacement

Il existe plusieurs méthodes de déplacement parmi lesquelles la méthode des ensembles de niveaux se distingue. Elle utilise un schéma décentré pour le calcul de la solution faible des équations de

la fonction distance, ce qui apporte une amélioration significative quant au traitement des intersections locales des courbes et des surfaces durant la propagation.

Cependant, le résultat du déplacement de courbes et surfaces dépend de plusieurs facteurs, et on réinitialiser la fonction des ensembles de niveau au cours des calculs pour maintenir la précision de la représention des ensembles de niveau. Ces exigences augmentent le temps de calcul et alourdissent la mise en oeuvre.

Dans la présente méthode on pose le problème de propagation de front comme la solution d'un problème à valeurs aux frontières, où le front à être déplacé est le niveau zéro de l'ensemble de niveaux et représente la condition frontière. Le front original et sa propagation sont implicitement compris dans les équations. Les courbes et surfaces d'iso-ϕ qui correspondent à la valeur de la distance du déplacement représentent les courbes et surfaces deplacées et doivent être extraites du domaine de calcul.

$$\begin{cases} \nabla\phi \cdot \nabla\phi = 1 \\ \phi = 0 \quad if\ \vec{P} \in \Gamma \end{cases} \tag{1}$$

où ϕ est la distance Euclidienne minimale entre le front (Γ) qui doit être propagé et P, un point arbitraire dans l'espace de calcul Ω.

La méthode présente possède tous les avantages de la méthode des ensembles de niveaux, c-à-d une façon naturelle et précise de suivre des coins aigus et des cornes, ainsi que la maîtrise de changements topologiques de fusion ou séparation de surfaces. Ceci est possible parce que ces deux méthodes reposent sur la solution avec viscosité artificielle des équations différentielles associées, afin de garantir une solution unique qui vérifie la condition d'entropie. De plus, la méthode présente est implicite et par conséquent l'approximation n'est pas sujette aux conditions de stabilité de CFL, ce qui n'est pas le cas de la méthodes des ensemble de niveaux.

La distinction entre la méthode de l'équation de distance de déplacement et la méthode des ensembles de niveaux est qu'elles approchent les problèmes de propagation de fronts d'un point de vue stationnaire et dynamique, respectivement. Dans la méthode des ensembles de niveaux, la propagation du front est traitée comme un problème à valeurs initiales où la valeur des fonctions de niveaux ϕ sont connues initialement. Cette fonction ϕ évolue dans le temps de manière que le niveau zéro de l'ensemble de niveaux est toujours identifié ave l'interface de propagation. Le niveau zéro de l'ensemble doit être extrait du domaine de calcul.

Le travail présenté utilise la méthode de balayage rapide (FSM), parce que comparée à la méthode d'avance rapide (FMM) (1) elle permet une efficacité élevée ($\Omega(M)$) comparé à ($\Omega(MlogM)$); (2) sa mise en oeuvre est directe et plus facile puisque sa structure de données est un tableau, comparé à une pile; et (3) des algorithmes de parallèlisation peuvent être appliqués directement.

La méthode de balayage rapide a été mise en oeuvre en 2d et validée avec les résultats de Zhao (Zhao (2005)). De plus, deux cas analytiques simples en 3d ont été utilisés afin d'établir la précision de ϕ pour l'algorithme proposé. Ces essais sont réalisés pour une sphère, centrée à (0,0,0) et de rayon $r = 0.15$, avec un référentiel cartésien de 0.5x0.5x0.5, et un cube centré à (0,0,0) et de dimension 0.3x0.3x0.3, avec un référentiel cartésien de 0.5x0.5x0.5. Dans ces essais, la solution exacte est connue, de sorte que le comportement de la convergence est facilement évalué.

Selon Zhao (Zhao (2005)), l'erreur du schéma ne dépend que du pas de la maille. Pour la sphère, le front original est spécifié comme des points isolés, et les valeurs de ϕ aux noeuds frontières sont initialisées avec les distances exactes de la surface de la sphère. Pour le cube, la discrétisation de la géométrie du front est spécifiée, et le comportement de deux normes (L_2, L_∞) est étudié en fonction de la taille de la maille.

Le Théorème 4.1 de Zhao (2005) énonce que pour un seul point Γ, la solution numérique de la

méthode de balayage rapide converge en 2^n balayages en R^n, où Γ indique l'ensemble de données à être propagées, et n est la dimension de l'espace. Ce théorème est démonté dans Zhao (2005). Dans le travail présent, on utilise directement le Théorème 4.1, et les résultats présentés montrent que tous les calculs sont obtenus avec 8 balayages pour des domains 3d. La convergence n'est pas influencée par le pas de la maille, mais seulement par la dimension de l'espace.

Application 1 - Construction de surface déplacée approximative

La méthode de déplacement développée a été appliquée avec succès pour construire des surfaces déplacées où on obtient une paramétrisation consistante entre la surface originale et son déplacement. Ceci est réalisé par une combinaison des techniques traditionnelles de déplacement direct et l'approche présente basée sur l'équation Eikonale.

La différence principale entre la méthode présente et les méthodes traditionnelles de déplacement se trouve dans la manière de calculer la direction normale. Dans le méthode de déplacement direct, les directions normales locales sont calculées avec des informations géométriques, telles que les positions des points voisins situés sur Γ. L'idée centrale de la méthode est donc de propager chaque point discrétisé de la géométrie originale du front Γ le long de sa direction normale dans l'espace ϕ plutôt que dans l'espace géométrique. On obtient alors une méthodologie qui en plus de résoudre le problème des auto-intersections, assure également un lien paramétrique entre la surface originale et son déplacement dans la plupart des cas.

Ces surfaces déplacées sont valides aussi bien pour des représentations discrètes que continues, tant que l'information paramètrique est disponible pour chaque type de surface. Deux type de données peuvent être utilisées:

Données polygonales discrètes Dans ce cas, la description géométrique est représentée par une surface en mosaique (c-à-d par triangles ou quadrangles), la méthodologie de propagation est validée en termes de précision et d'efficacité.

Représentation NURBS continue Les points de contrôle de la NURB sont propagés, et le résultat est validé.

Dans ce travail toutes les courbes et surfaces sont représentées avec des NURBS qui peuvent être discrétisées à une précisoin arbitraire. La géométrie donnée Γ est discrétisée en segments (2d) ou en triangles (3d), et se situe à l'intérieur d'un domaine Ω. Le déplacement est réalisé par la propagation de tous les points situés sur Γ le long de la direction normale locale en utilisant l'équation suivante:

$$\vec{x}^{n+1} = \vec{x}^{n} + \vec{F}dt \tag{2}$$

où F est la vitesse de propagation unitaire.

La distance de déplacement ϕ, la distance Euclidienne minimale, est obtenue pour chaque noeud de la grille cartésienne en fonction des coordonnées spatiales tel qu'établi à la Section 4.3 et en utilisant l'algorithme décrit à la Section 4.4. Ce champ est ensuite utilisé pour évaluer la valeur de la vitesse de propagation unitaire locale $\nabla\phi/|\nabla\phi|$ à chaque noeud sur Γ.

Après la propagation, les valeurs finales $\vec{x}^{n+1}$ sont reliées séquentiellement afin de construire Γ_p, le front propagé, selon leur connectivité originale. Ainsi la consistance de la paramétrisation entre Γ, la surface originale, et Γ_p, son déplacement, est préservée, c'est-à-dire qu'il y a une correspondance univoque entre tous les points de Γ et de Γ_p. Cependant lors de la collision entre deux fronts cette information paramétrique ne peut être maintenue.

Les trois étapes essentielles de cette métode sont:

1. Le domaine Ω est discrétisé sur une grille cartésienne;
2. Le champ ϕ est calculé à chaque noeud de cette grille en utilisant l'algorithme de balayage rapide;
3. Chaque noeud sur Γ est déplacé le long de la direction normale locale de la valeur de la distance de déplacement.

Application 2 - Génération du maillage pariétal

La méthode proposée est utilisée pour générer des mailles de rapport de forme élevé dans le voisinage des parois pour des simulations d'écoulements à grand nombre de Reynolds. Les problèmes où les effets de parois dominent sont caractérisés par de forts gradients dans la direction orthogonale à la paroi comparés aux autres directions. Ceci nécessite une taille d'éléments minimale dans cette direction. L'alignement d'éléments anisotropes est essentiel pour cerner les caractéristiques de l'écoulement, avec une petite taille dans la direction de forts gradients et une plus grande dans les autres directions.

La méthode proposée est conçue pour engendrer efficacement et de façon fiable des maillages anisotropes semi-structurés près de frontières dans des domaines arbitrairement complexes à partir d'un maillage surfacique de triangles. Dabord, une décomposition en blocs, issus des frontières du domaine, est engendrée, conforme par les faces contiguës. Ensuite, à l'intérieur de chaque bloc on construit des maillages semi-structurés, avec un redistribution utilisant un opérateur Laplace-Beltrami aux interfaces des blocs, et un opérateur Laplace à l'intérieur des blocs.

Dans ce travail on utilise la technique de propagation de front à cause de sa capacité de contrôle de l'orthogonalité et de la taille des éléments. Les intersections des surfaces par elles-mêmes sont

évitées de façon naturelle et automatique tel que présenté en détail au Chapitre 4. Le dilemme entre le choix d'hexahèdres ou de tétrahèdres est résolu par l'utilisation de ces deux familles d'éléments: des éléments prismatiques étirés dans les régons pariétales, où des phénomènes de paroi (couches limites) sont attendus, et des éléments tétrahèdriques isotropes ailleurs dans le domaine. Les éléments prismatiques sont composés de faces triangulaires dans la direction tangengentielle à la paroi, et de faces quadrilatères dans la direction perpendiculaire. Ils combinent la flexibilité géométrique des éléments nonstructurés ainsi que l'orthogonalité et l'élancement des maillages structurés.

Le éléments tétrahèdriques semblent appropriés dans le coeur du domaine à cause de l'irrégularité des formes. Les faces triangulaires des tétrahèdres s'adaptent aux faces correspondantes des prismes, formant un maillage globalement conforme.

Le processus de génération de maillage est résumé en cinq étapes principales:

1. les surfaces de la frontière Γ du domaine de calcul sont discrétisées en faces triangulaires;
2. le champs de distance est calculé;
3. les noeuds situés sur Γ sont déplacés, et les interfaces de la décomposition multi-bloc sont engendrés;
4. les éléments anisotropes sont redistribués à l'intérieur de chaque blocs;
5. l'extérieur de la couche est discrétisé avec des tétrahèdres.

La démarche décrite est appliquée au cas industriel du modèle d'un aspirateur 3d d'une turbine, pour des configurations ouverte et fermée. La qualité (répartition volumique et élancement) du maillage résultant est mesuré montrant que la qualité est acceptable.

Application 3 - Calcul d'écoulements fluides

Le but de cette application est de valider la qualité et l'efficacité d'une nouvelle stratégie de génération de maillage hybride appliquée à l'écoulement dans un modèle de valve sphérique. La qualité du maillage est mesurée sous trois aspects: (1) la distribution volumique; (2) le rapport de taille et (3) l'élancement normalisé. Ces résultats montrent que la qualité du maillage est acceptable, et les solutions convergées de l'écoulement fluide sont illustrées à la Section 7.4.

Le but de la simulation de l'écoulement fluide est de déterminer les champs de vitesse et de pression à la sortie du modèle 3d pour valider la méthodologie de génération de maillage proposée. Les conditions de l'écoulement sont en position demi-ouverte et en régime stationnaire et instationnaire. Topologiquement, la géométrie est très simple et équivalente à un cylindre. Géométriquement, la configuration présente des formes convexes et concaves. La partie la plus difficile est de calculer correctement la direction normale aux quatre points singuliers tel qu'illustré à la Figure 7.2 où quatre surfaces s'intersectent et partagent un même point. La méthode traditionnelle de calcul de la normale utilise l'information géométrique autour du point à être propagé, c-à-d la moyenne pondérée des vecteurs normaux aux surfaces voisines. Dans de tels cas, la définition de vecteur normal devient ambiguë lorsque calculée de la sorte.

On illustre les étapes du processus de maillage, et comment les problèmes identifiés sont résolus par la méthodologie proposée, particulièrement en ce qui a trait au comportement dans les coins aigus, et les singularités. Les champs de vitesse et de pression sont illustrés à la Section 7.4.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOTATIONS AND SYMBOLS

NURBS	Non Uniform Rational B-Spline
CFL	Courant-Friedrichs-Levy condition. It is an important necessary but not sufficient condition for the stability of numerical methods
STL	Standard Template Library
IOM	Indirect Offset Method
DOM	Direct Offset Method
PDE	Partial Differential Equation
FSM	Fast Sweeping Method
FMM	Fast Marching Method
CAD	Computer Aided Design
STL	Stereolithography

Greek Characters

Ω	the computational complexity
Γ	the front to be propagated.
ϕ	the minimum Euclidean distance from any point in the computational domain to the Cartesian grid point.

CHAPTER 1

MOTIVATION

1.1 Background

The quality of numerical solutions are critically dependent on various discretization techniques as they apply to the geometry of the domain as well as to the governing equations. More specifically regarding spatial discretization, numerical methods rely for accuracy and efficiency on the shape and distribution of the elements used in the grid generation processes.

Grid generation has evolved rapidly over the last two decades, resulting in a well established methodology to generate structured or unstructured grids for an entire domain, known as *single-block grids*. This methodology includes three elements:

- *Geometric Modeling*, which is the foundation for the geometric representation of the boundaries;
- *Grid Generation*, which discretizes the physical domain using specialized elements (hexahedra, tetrahedra or prisms) and appropriate algorithms (structured, unstructured grids);
- *Grid Adaptation*, which improves the mesh quality through some combination of redistribution, refinement, and topological re-connections of the elements.

While the single-block strategy (e.g. all tetrahedra mesh) has achieved a high level of success to generate grids in three dimensions even for highly complex shapes when solver requirements

are not too stringent, further improvements are still expected. New approaches are still required to overcome the difficulties encountered as more complex objects are modeled and analyzed in highly complex physics such as high Reynolds turbulent flow. Specifically these difficulties are related to the local control of the grid characteristics to capture specific features related to the physics of the problem, and to the automation issue as it applies to an engineering production environment.

A promising strategy to overcome these and associated problems is to use generalized *multi-block grids*, a technique which reduces a single geometrically complex grid generation problem into several grid generation problems in a set of simpler components. Multi-block strategies consist of a partition of the physical domain into a few overlapped sub-domains or non-overlapped contiguous sub-domains, called *overset grids* or *hybrid grids*, respectively. The resulting blocks may be considered as the cells of a coarse mesh, and at the individual block or sub-domain level, the mesh can be locally structured or unstructured, but it is globally unstructured when viewed as a collection of blocks.

The benefits of using multi-block grids are:

- Local grids in each block are much easier to generate than a single global grid for the whole domain, as the geometry of each block is generally simpler;
- More flexibility for grid adaptation and modification, since the regeneration of the local grids is applied within the relevant blocks rather than to the entire domain;
- Flexible physical modeling for problems which are heterogeneous relative to some of the physical quantities, allowing different mathematical models in different zones of the domain;
- Processing may be embedded in the mesh generation algorithms whereby different blocks can be assigned to different processors, resulting in an improved efficiency, especially for large problems.

1.2 Goals and Objectives

As discussed before, to improve the grid generation process for complex geometries, it is necessary to first subdivide complex domains into blocks, and then apply a local meshing process to each block. Since multi-block strategies prevail over single-block strategies in mesh generation processes, it is greatly promoted in industrial applications and is employed in this research, as the former can combine several single-block strategies for specific sub-domains.

There are two major types of multi-block strategies: *hybrid grids* and *overset grids*. According to the topological arrangement of interfaces between blocks, multi-block grid generation strategies can be classified as: *overlaid Grids*, in which the interfaces of blocks are overlapped and *Hybrid Grids*, in which the interfaces of blocks are non-overlapped.

In overlaid grid strategies, separate grids are generated about each boundary component, and these separate grids are simply overlaid on a background grid and perhaps on each other in a hierarchy.

Overlaid grids are also known as *Chimera*, or *overset* methods. The advantages of *overlaid* grids are: (1) simplicity in grid generation since the various grids are generated separately, (2) local body-fitted properties are preserved since separate boundary-conforming structured grids are generated for each component of a complex configuration, and (3) data is communicated between the various component grids by interpolation.

The hybrid grid strategy decomposes the domain into blocks which can be viewed as unstructured grids globally, and then structured or unstructured grids are generated in each block.

The combination of grids of different types not only allows the benefits of structured and unstructured grids to be attained simultaneously, but also allows high grid quality to be achieved

throughout the domain due to the appropriate use of each element type.

This research proposes to use a hybrid grid strategy rather than the overlaid grid strategy. The reasons are given below:

1. Overlaid grids are suitable to simulate problems which have relative motions between components of multiple-body configurations, e.g., moving stores, flaps. However, these are not attractive features to this research.

2. In hybrid grids, the mesh points at interfaces of blocks are matched to each other. This is an attractive feature for this research especially at boundary areas, where different types of mesh cells are required.

3. One of the defect for overlaid grids is that interpolations are required at the overlapped mesh area for overlaid grids, which severely decreases the accuracy of the solution.

The aim of the present work is to develop a new hybrid grid generation method. The bottleneck in the hybrid grid strategy is the domain decomposition and, in particular, its automation in an engineering context. This research investigates such domain partitioning methodologies and proposes a hybrid meshing strategy and related techniques capable of handling highly anisotropic features as they occur in wall dominated phenomena such as viscous layers in aerodynamic applications. This type of physical problem exhibits strong gradients in directions orthogonal to the wall compared to the other directions. To capture the directionality of the phenomena, highly anisotropic meshes are required in the vicinity of solid walls. Away from the boundaries, isotropic elements can be used. Hence, the necessity to partition the domain into a wall layer or skin region, and the remainder of the domain.

Current hybrid meshing methodologies will be extended with the following specific objectives:

Enhancement of the feature capturing capabilities :

The ability to simulate high-gradient features for viscous flows in boundary layers depends on the capability of the mesh to adequately locate these regions and to assign an appropriate mesh type and density. This research aims to develop a new boundary layer mesh generation technique to overcome the major weaknesses in current approaches.

Increase of the level of automation :

Although many hybrid meshing techniques have been presented, the automatic meshing of an arbitrarily complex domain is still a very significant bottleneck of the mesh generation process. One important objective of this research is to attempt to develop an algorithm with a high level of automation which minimizes user intervention.

Improvement of the computational efficiency :

The use of the Fast Sweeping Method increases computational efficiency, since its computational complexity is $\Omega(N)$, where N is the total grid number of the computational domain. Also, parallelized algorithms can be applied to this numerical scheme.

1.3 Research Contributions

Many domain decomposition methods have been presented that aim to apply hybrid meshing strategies to boundary layer mesh generation. This research proposes an efficient way to decompose the domain in the vicinity of the geometric boundaries where boundary or skin meshes are required. The proposed method is based on a front propagation technique which brings two major advantages compared with the traditional techniques:

A new domain decomposition technique :

A new domain decomposition method is developed based on the offset technique, this technique formulates the domain decomposition problem as an offset problem and its solution is obtained using the Eikonal equations. This technique can be applied to solve boundary dominated problems, for example, viscous flow, heat transfer, etc.

A simplified mathematical model :

A new simplified mathematical model, the *Offset Distance Equation*, which is a variation of the Eikonal equation, is proposed to solve the front propagation problem. An extended validation of the *fast sweeping numerical scheme* to 3D applications with smooth surfaces is performed. This part of the work is summarized in the paper (Wang *et al.* (to be published)), submitted to the International Journal for Numerical Methods in Engineering.

An improved normal direction calculation :

Normal directions are represented using the numerical solution of the Eikonal equation, which avoids self-intersections from occurring during propagation. The details about preventing self-intersection is discussed in Chapter 5.

A new offset surface construction method :

As an extension, this approach has been applied to the propagation of geometrically complex fronts, i.e. convex and concave boundaries in multiply-connected domains. Specifically, different techniques have been developed to account for the construction of such offset surfaces in diverse types of topologies such as permeable surfaces for inlet/outlet boundary conditions.

The proposed method has been successfully applied to different applications, for example, surface offset construction, boundary meshing process, and fluid flow analysis. These works are published in Wang *et al.* (June 2005), Wang *et al.* (September 2005) and Wang *et al.* (September 2006) respectively.

1.4 Thesis Organization

The organization of this thesis is as follows: a literature review for domain decomposition techniques is presented in Chapter 2. As the geometric complexity of configurations in industrial applications increases, mesh processing is increasingly being replaced by multi-block strategies, where partitioning techniques play a key role. Domain decomposition is a topological process which aims to simplify complex geometries by subdividing the domain into smaller and simpler regions called blocks. This chapter goes through the major historical improvements of this technique.

The domain decomposition technique developed in this work is a direct inspiration from front offset technique, thus a literature review for boundary offset techniques is presented separately in Chapter 3. Since there is great incentive to use anisotropic grids over boundaries in viscous flow simulation, where the boundary layer requires very small spacing out from the wall. The major problems in boundary offset encountered in classical front propagation methods are: 1), detecting and resolving self-intersections, and 2), preserving the one-to-one parameterizations mapping from original surface to the offset surface. This chapter introduces the major methods currently available to solve offset problems and their efforts to avoid the problems mentioned above.

Chapter 4 gives a brief derivation of the *Offset Distance Equation* model, and explains why the proposed model can solve the front propagation problem. The fast sweeping numerical scheme to

solve this equation is then briefly summarized; validations using smooth 3D surfaces are presented, which confirm the convergence theorems of the fast sweeping method. Finally, some 2D and 3D propagation results are presented, which show that the presented method can be used for solving propagation problems.

Chapter 5 illustrates the proposed front propagation methodology, to construct offset surfaces and transport the parameterization of the original surface to the propagated surface. This method, based on the solution of the equation derived from the Eikonal equation, comprises three essential steps : (1) construct a uniform Cartesian grid fully covering the domain of interest, consisting of discretized curves and surfaces; (2) solve the Eikonal equation using the fast sweeping numerical scheme on the Cartesian grid, and, (3) march in space from the discretized original geometric front points, along their local normal directions, by iteratively solving a propagation equation, using a fourth-order Runge-Kutta scheme. The most attractive features of this technique are that in the absence of blocks, it can exactly map the parameterization information from the original front to the offset front, and local normal directions are represented by the ϕ field , thus they do not intersect each other during propagation. This technique can also be used to offset the control point of NURBS represented curves and surfaces, and to generate meshes in boundary layer areas. Numerical experiments are presented to demonstrate the accuracy and efficiency of the method.

In Chapter 6, the proposed method is applied to automatic boundary block meshing of engineering configurations, with through-flow boundaries to illustrate the use of a permeable front. The decomposition approach proceeds by explicitly constructing offset surfaces to boundaries, which are then used to topologically subdivide the domain into simple regions that are meshed independently. A key contribution of this approach lies in the method used to construct the boundary offset surfaces and construct the structured mesh in each block. A mixed method for the solution of the offset distance equation, derived from the Eikonal equation, is used, that alleviates local and global self-

intersection problems during offset surface construction, while allowing to maintain a parametric relationship between the original surfaces and their corresponding offset. Mesh results in actual industrial geometries are used to demonstrate the validity of the method.

A robust and automated approach to generate unstructured hybrid grids comprised of prismatic and tetrahedral elements for viscous flow computations is presented in Chapter 7. The hybrid mesh generation starts from a triangulated surface mesh. The prismatic elements are extruded based on the proposed algorithm to generate anisotropic elements at boundaries, and finally the isotropic tetrahedral grids are generated to fill the rest of the domain. The presented hybrid meshing algorithm was validated using a ball valve model under both steady and unsteady conditions.

Finally, conclusions and future research directions are discussed in Chapter 8.

CHAPTER 2

DOMAIN DECOMPOSITION

Early mesh generation techniques were based on single-block strategies. As the geometric complexity of configurations in industrial applications increased, this has proved to be inadequate. It is increasingly being replaced by multi-block strategies, where the partitioning technique plays a key role.

The interest of this research lies in the application of the front propagation techniques, as a multi-block domain decomposition tool and in the investigation of its effectiveness for partitionning a global domain into sub-domains, particularly in the vicinity of solid boundary areas.

This Chapter will briefly review domain decomposition in general. Specifically, this will emphasize its application as a domain decomposition technique in an overall multi-block grid generation methodology. These will be reviewed and critically compared in the light of the objectives of this work.

2.1 Introduction

Domain decomposition is a topological process which aims to simplify complex geometries by subdividing the domain into smaller and simpler regions called blocks. The development of this approach is motivated by the need to simplify the task of meshing complex geometries in order to make such problems more manageable. In addition, it allows the apllication of different mesh

types and/or generation techniques in different regions (or blocks). Finally, parallel algorithms can be embedded in the mesh generation algorithms where different blocks can be assigned to different processors, thus the computational efficiency of the program can be greatly improved.

2.2 Unstructured method

Some researchers have employed unstructured triangulation methods to first generate triangular sub-domains by applying the Delaunay technique (Bergman (1990); Cordova (1992)), and then transform the triangles into quadrilaterals by removing of the appropriate edges or subdividing each triangle into three quadrilaterals that were used as blocks (Bergman (1990)). Schonfeld and Weinerfelt (1991) decomposed the domain directly into quadrilaterals using an advancing-front technique. Stewart (1992a) and Stewart (1992b) proposed an approach in 2D based on a small set of search rules to drive directional probing from the boundary to search for an appropriate block decomposition, in analogy with balloons inflating against each other: a coarse approximation to the outer boundary of the region was obtained. Kim and Eberhardt (1995) used advancing front to automate block construction by generating coarse hexahedra cells for 3D problems. The distance and general direction of advancement were user controlled. The method tended to create a large number of relatively small blocks, since all are full-face matched.

2.3 Prime Rectangle Method

A new domain decomposition algorithm has been presented in Piperni and Camarero (2003), which aims for the automation of structured grid generation in multiply-connected domains. This domain decomposition technique has many new features: *(1)* it can decompose an arbitrary complex do-

main automatically without human intervention, *(2)* it generates overlaid blocks which is an important advantage in the mesh optimizing process, and *(3)* it guarantees a unique decomposition topology for the physical domain once the geometric and topological information are specified.

The basic idea of this method is to first subdivide the physical domain into non-overlapped blocks to generate algebraic meshes in each block; and then decompose the physical domain into overlapped blocks to optimize the algebraic meshes by an elliptic method through successive scanning of the overlapped blocks. The whole meshing process is carried automatically, resulting in a globally structured mesh.

Disjoint Rectangle

If a rectilinear polygon[1] consists of a set of minimum number of non-overlapped rectangles, then this set of rectangles is called disjoint rectangles.

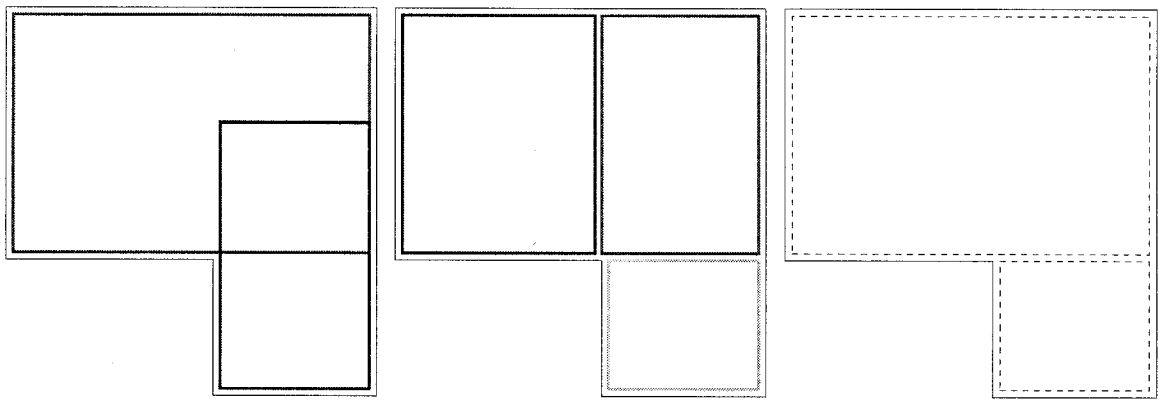
Figure 2.1(a) illustrates a rectilinear polygon covering by non-disjoint rectangles, since the set of rectangles are overlapped; Figure 2.1(b) illustrates a rectilinear polygon covering by non-disjoint rectangles, since the number of decomposed rectangles is not minimum. Figure 2.1(c) illustrates a rectilinear polygon covering by disjoint rectangles, since it meets all the conditions of disjoint rectangles.

Prime Rectangle

A *Prime Rectangle* is a rectangle of maximum size for a given area of the region. It is such that a part of each of the four sides of a prime rectangle must touch a domain boundary. The relationship of a prime rectangle P and any rectangle P' in a given figure F is such that:

1. $P \subseteq F$;

[1] A *rectilinear polygon* is a polygon whose sides are either perpendicular or parallel to each other.



(a) Non-disjoint rectangles (b) Non-disjoint rectangles (c) Disjoint rectangles

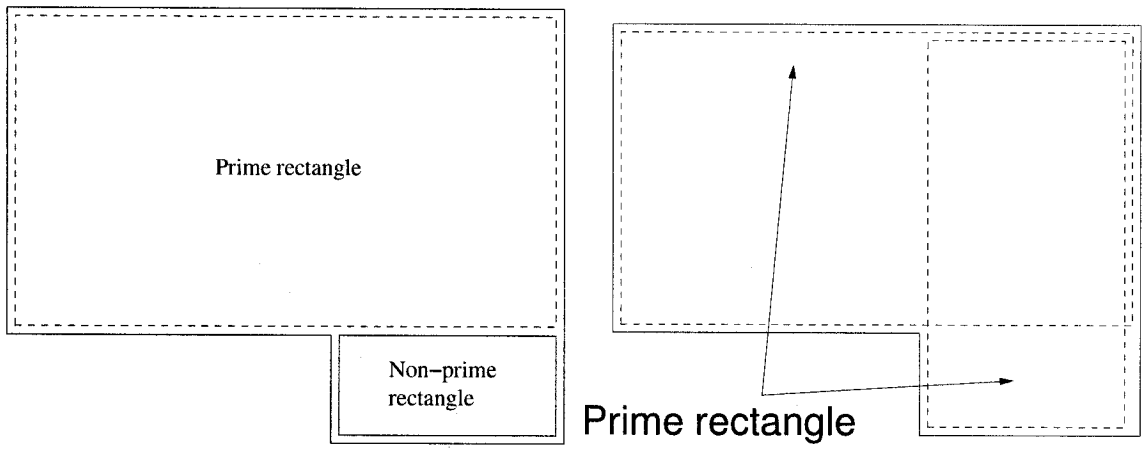
FIGURE 2.1 *Disjoint* and *non-disjoint* rectangles

2. no $P' \subseteq F$ exists, such that $P \subset P'$.

A rectilinear polygon may consist of overlapped prime rectangles. Consider the problem of subdividing a given non-simple rectangular domain, the dash-line rectangle in Figure 2.2(a) is a prime rectangle, while the solid-line rectangle is not a prime rectangle as its upper side can be extended along its upper direction to touch the upper boundary. Figure 2.2(b) is covered by prime rectangles.

The application of this method consists of the following steps:

Domain geometric and topological configuration Domain configuration is a critical step since it leads to a unique domain decomposition. *Domain geometric configuration* is performed by inputting the geometry boundaries and the outer boundaries. *Domain topological configuration* includes specifying the boundary conditions (or mappings), i.e. the values of the mesh indices on the geometry and outer boundaries. A rectilinear polygon will be constructed after this configuration.



(a) Non-prime rectangle (b) Prime rectangle

FIGURE 2.2 *Prime* and *non-prime* rectangle

Domain Decomposition Sub-domains are generated by: *(1)* decomposing the rectilinear polygon into disjoint rectangles according to the algorithm of (Lipski *et al.* (1979))[2], and *(2)* mapping these disjoint rectangles to the physical domain to generate non-overlapped blocks.

Mesh generation Initial meshes are generated by algebraic method in individual blocks in the physical domain.

Mesh optimization The rectilinear polygon is further decomposed into prime rectangles according to the algorithm of (Lodi *et al.* (1979))[3]; and then these prime rectangles are mapped into physical domain to generate overlapping blocks. The initial algebraic meshes are optimized successively in the overlapping blocks by elliptic method (Spekreijse (1995)) through an iterative loop. The resulting mesh is a global non-overlapped structured mesh.

[2]It can decompose an arbitrary rectilinear polygon into disjoint rectangles automatically.
[3]It can decompose an arbitrary rectilinear polygon into prime rectangles automatically.

2.4 Boundary-based domain decomposition

Guibault (1998) proposed a domain decomposition method, based on a direct propagation of the boundaries, aimed at generating multiple blocks around geometric boundary areas. Topologically, each block is equivalent to a cube template. The blocks are obtained by first generating surface grids on the boundaries to be offset, then propagating these surface grid points along their normal directions. Finally, the relative topological connectivities are constructed by sequentially connecting propagated points into edges, faces and volumes. The input and output files include both geometric information, (points, curves and surfaces), and topological information (vertices, edges, faces and volumes).

2.5 Discussion

Among all the reviewed blocking methods discussed in previous sections, only the *Unstructured Methods*, the *Prime Rectangle Method* and the *Boundary-based Domain Decomposition Method* have effectively achieved a high level of automation. They can accept geometric and topological information as input, and output the resulting mesh with minimum user intervention.

Unstructured Domain Decomposition

Although unstructured domain decomposition algorithms have been developed with a high level of automation, these have two limitations:

(1) They do not allow globally structured grid generation, while local structured grids are possible to be generated internally for each block, and

(2) once the interfaces between blocks are generated, it is difficult to float the block interfaces.

Prime Rectangle Method

Because the physical domain is decomposed into overlapped blocks, the prime rectangle method presents some advantages over the unstructured method:

(1) It makes data interchanges possible during the optimization process,

(2) It allows free-floating interfaces between blocks in the optimization process, and

(3) The overlapped blocks preserve the properties of local body-fitted boundaries.

However, this method requires expertise from the user in the specification of the mesh topology which is cast as a boundary value problem. The resulting meshes are highly dependent on the topological definition which is prescribed by the user.

Boundary-based Domain Decomposition Method

Both the unstructured and the prime rectangle decomposition techniques are suitable for partitioning an entire domain into several sub-domains. This kind of technique can be viewed as a global domain decomposition technique since it is very effective to decompose the entire domain into sub-domains. But they do not present any potential to generate thin layers in the vicinity of boundaries, in which high-aspect-ratio grids are required for capturing high gradients.

Boundary-based domain decomposition methods generate a skin around the geometric boundaries which is the desired domain decomposition technique for this research. The advantage of this method is that it enforces parametric consistency between the original boundary, and its offset

counterpart. All the points located on the original front are propagated along their local normal directions, thus all the points on the original front and its offset share a one-to-one parametric mapping. The weakness of this method is that self-intersection eliminating algorithms are difficult to implement and may fail for severely concave regions in 3D.

* * *

The following chapter will explore another type of techniques for thin skin generation, local domain decomposition technique, and specifically concentrate on reviewing front propagation techniques.

CHAPTER 3

BOUNDARY OFFSET

Computing offset curves and surfaces is a technique to track the evolution of a front propagating along its normal vector field. This has diverse engineering applications, including, for instance, tracking the motion of interfaces in fluid mechanics (Smereka and Sethian (2003)); tool path generation in numerical control machining (Park and Chung (2003), Lee (2003)); image denoising and enhancement in computer graphics and vision (Malladi and Sethian (1996), Osher (2003)); and gap closing in computer-aided design (Sethian (1999)).

In the present work we will apply the generation of an offset boundary to create a thin layer in the vicinity of solid boundaries (Pirzadeh (1993), Guibault (1998)). This procedure is part of the global strategy to partition the domain into regions of highly stretched grid elements close to boundaries, and regions of isotropic grid elements elsewhere.

3.1 Offset Problem Description

Figure 3.1 illustrates the boundary offsetting problem, where $\gamma(\vec{x}, t_0)$ is the surface generated by moving $\gamma(\vec{t_0})$ along its normal vector field for a given parameter t. The position of $\gamma(\vec{x}, t)$, as it evolves in the normal direction is such that this motion ignores displacement along its tangential direction. It can be expressed as:

$$\gamma(\vec{x}, t) = \gamma(\vec{x}, t_0) + \vec{n}dt \tag{3.1}$$

where $\vec{n}$ is the normal vector and dt is the time step.

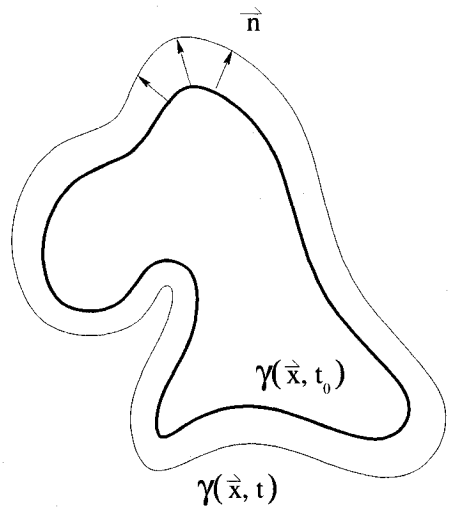


FIGURE 3.1 Boundary Offset Problem

The major problem encountered in classical front propagation methods is detecting and resolving self-intersections. Two situations can arise:

- a local warping may occur when the offset distance is greater than the local curvature radius in concave regions as illustrated in Fig. 3.2, when the curve has a discontinuity.
- a global self-intersection may occur when the distance between two distinct points on the curve or surface reaches a local minimum as illustrated in Fig. 3.3.

A variety of contributions deal with the computation of curve and surface offsets. They can be classified in two types: *direct offset methods (DOM)*, which propagate curves or surfaces directly based on original geometric data; and *indirect offset methods (IOM)*, which cast curve or surface offset problems into a set of partial differential equations (PDE), with the numerical solution to

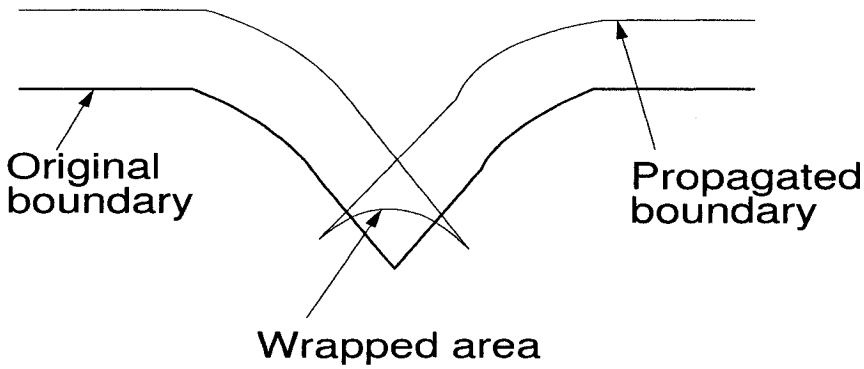


FIGURE 3.2 Local wrapped area

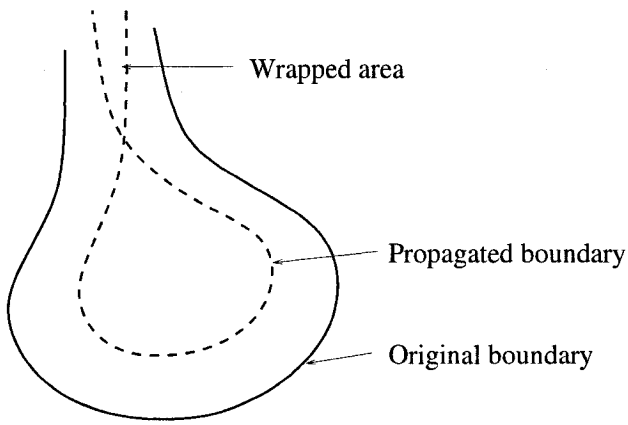


FIGURE 3.3 Global wrapped area

these being the offset curves or surfaces. The capacity to effectively eliminate self-intersections is an important criterion and we will review and evaluate each method in that context.

3.2 Direct Offset Methods

Many variants of the direct propagating methods which make use of Eqn. 3.1 have been proposed. The common point of these methods is that they construct offset surfaces based simply on geometric information of the given surface to be propagated.

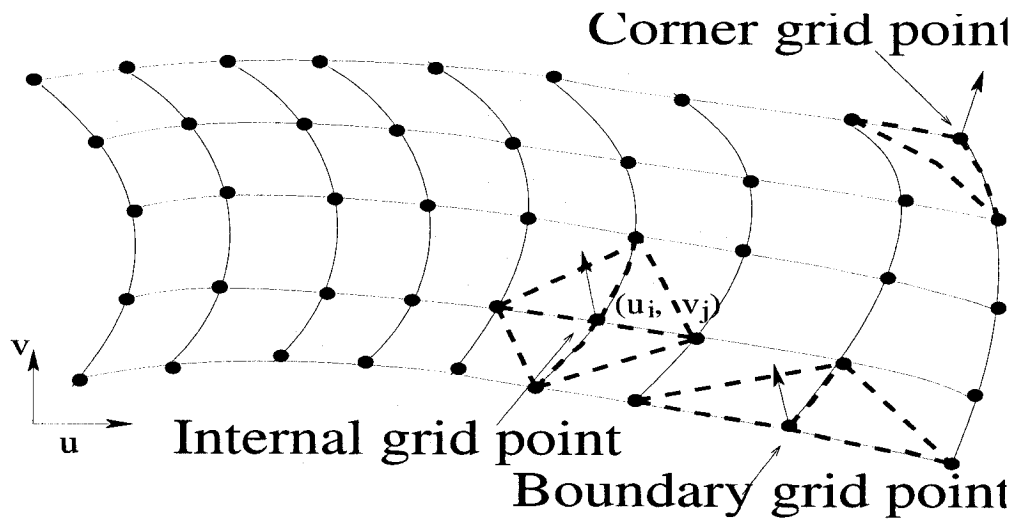


FIGURE 3.4 Direct offset method - quadralitrial discretization

The basic idea of direct offset method is shown in Figure 3.4. A surface grid is generated by first defining triangular or quadrilateral elements in surface patches from the geometric representation. Surface normal vectors are determined at each mesh point by averaging the unit normal vectors of the faces sharing the point and then smoothing the resulting vectors by a Laplacian operation. This surface is advanced as one layer by connecting the surface grid points and the tip of the surface vectors. The spacing for each layer of cells is determined by a geometric function.

3.2.1 Advancing-Layers Method

The direct offset technique was applied to grid generation by Pirzadeh (1993, 1994) and Pirzadeh (1996). Known as the advancing-layers method, it is a marching strategy where layers are advanced according to Eqn. 3.1. The aim of this method was to generate highly stretched cells to resolve the boundary layer in three dimensional problems. In this method, a surface grid is generated on the boundary as line segments in two dimensions or surface patches in three dimensions from the geometric representation. Grid points are then distributed along the lines to form an initial front in two dimensions. In three dimensions, surface patches are triangulated to construct the surface grid.

Surface normal vectors are then determined at each mesh point by averaging the unit normal vectors of the faces sharing the point, and then smoothing the resulting vectors by a Laplacian operation. Finally, this surface is advanced as one layer by connecting the initial surface grid points and the tip of the averaged surface vectors. The spacing for each layer of cells is determined by a specified geometric function. In this method, the front and the grid are generated simultaneously. It starts from a surface grid rather than from a geometric description of the surface itself.

One representative attempt to eliminate self-intersections is to simply stop the advancement of the front before self-intersections occur thus avoiding the problem. Based on a similar marching idea, Sullivan and Zhang (1997) presented a self-intersection detection and removal algorithm in 2D. This approach uses a normal offsetting technique to develop nodes followed by Delaunay triangulation with a mesh resolution function based on the physics and problem geometry. The method can handle multi-connected boundaries and multiple material regions.

3.2.2 NURBS offset method

The *NURBS direct offset method* aims to improve the boundary offset process using a formal representation of the boundaries based on a NURBS geometric representation (Coquillart (1987)). In this method, the offset curve is defined by a new control polygon where each new control point is the offset of the corresponding control point of the original curve in a direction given by the normal at the closest point of the curve. Guibault (1998) has given an improved algorithm to offset a NURBS B-Rep boundary. In this procedure, the surface grid generation is obtained from the NURBS representation where the boundaries are discretized along their u and v directions. The normal vectors for each surface point is computed by averaging the unit normal vectors of the faces sharing the point.

NURBS represented curve C is defined as:

$$\vec{C} = \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^{n(t)}}{\sum_{i=0}^{m} w_i N_i^{n(t)}} \tag{3.2}$$

n is the total number of the control points, the $\vec{P}_i$ is the ith control point, the w_i is the ith weight, and the $N_{i,p}(u)$ are the pth-degree B-spline basis functions defined on the nonperiodic (and nonuniform) knot vector U,

$$U = \{\underbrace{a, ..., a}_{p+1}, u_{p+1}, ..., u_{m-p-1}, \underbrace{b, ..., b}_{p+1}\} \tag{3.3}$$

The normal vectors for 3D curves and surfaces are calculated using the definition of differential

geometry (Farin (1997)).

$$\vec{n} = \frac{\vec{C}_u \times \vec{C}_{uu}}{||\vec{C}_u \times \vec{C}_{uu}||} \times \frac{\vec{C}_u}{||\vec{C}_u||} \tag{3.4}$$

where $\vec{C}_u$ and $\vec{C}_{uu}$ are the first and second derivatives of the curve with respect to the u-direction, respectively.

$$\vec{C}_u = \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u)' \sum_{i=1}^{m} w_i N_i^n(u)}{(\sum_{i=0}^{m} w_i N_i^n(u))^2} - \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u) \sum_{i=1}^{m} w_i N_i^n(u)'}{(\sum_{i=0}^{m} w_i N_i^n(u))^2} \tag{3.5}$$

$$\begin{aligned} \vec{C}_{uu} = & \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u)'' \sum_{i=1}^{m} w_i N_i^n(u)}{(\sum_{i=0}^{m} w_i N_i^n(u))^2} \\ & - \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u)' \sum_{i=1}^{m} w_i N_i^n(u) \sum_{i=0}^{m} w_i N_i^n(u)')}{2 * (\sum_{i=0}^{m} w_i N_i^n(u))^3} \\ & - \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u) \sum_{i=1}^{m} w_i N_i^n(u)''}{(\sum_{i=0}^{m} w_i N_i^n(u))^2} \\ & + \frac{\sum_{i=0}^{m} w_i \vec{P}_i N_i^n(u) \sum_{i=0}^{m} w_i N_i^n(u)' \sum_{i=1}^{m} w_i N_i^n(u)'}{2 * (\sum_{i=0}^{m} w_i N_i^n(u))^3} \end{aligned} \tag{3.6}$$

The offset surfaces are generated by moving each point along its normal vector according to a specified distance, and then linearly connecting the new grid points sequentially in the u direction for curves (u and v directions for surfaces) to construct propagated curves (bilinear patches for

surfaces).

In the 3D algorithm developed by Guibault (1998), one proceeds by first detecting tangled loops, etc., and then these are eliminated by redistributing the points located within the looped area (Figure 3.5). The interpolation formula is given below:

$$\vec{x}_{P_i'} = (1-r)\vec{x}_{P_0} + r\vec{x}_{P_{n+1}} \tag{3.7}$$

where, $\vec{x}_{P_i}$ denotes the position of point P_i which is to be deleted from the loop, $\vec{x}_{P_i'}$ denotes the new point position to be inserted which corresponds to the deleted point P_i; r is the interpolation ratio at i-*th* point. This approach requires considerable trial and error, and the resulting interfaces are not always satisfactory.

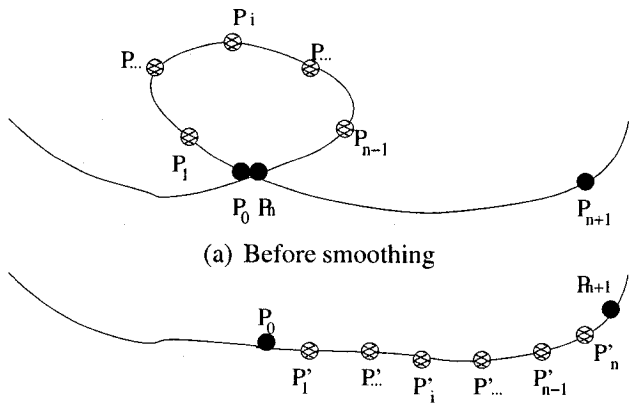


FIGURE 3.5 Smoothing algorithm

In the procedure described by Glimm *et al.* (2000), an algorithm is applied to resolve self-intersections by either re-triangulating the relative elements after removing unphysical surfaces, or reconstructing the interface within each rectangular grid block in which crossing is detected.

Other types of techniques to eliminate self-intersections use the properties of curves and surfaces, i.e. control points, derivatives, curvature etc. Blomgren (1981), Tiller and Hanson (1984), and Coquillart (1987) approached the problem by offsetting the control polygon for NURBS curves. Kulczycka and Nachman (2002) extended this idea to propagate surfaces by offsetting control points. Piegl and Tiller (1999) sampled offset curves and surfaces based on bounds on the second derivatives to avoid self-intersections. In the method developed by Sun *et al.* (2004), control points are repositioned to reduce local curvature in areas where local self-intersections may occur, while the rest of the control points remain unchanged. Farouki (1986) described an algorithm which first decomposes the original surfaces into parametric patches, and then uses Hermite interpolation to construct the offset surfaces.

3.2.3 Trimmed surface offset method

Kumar *et al.* (2003) proposed an algorithm to offset a trimmed NURBS surface (Please refer to Fig. (3.6)). In this development,

1. the underlying surface of the trimmed surface is propagated along the normals with the given offset distance;

2. the computed offset surface is extended in all four sides with a distance equal to the offset distance to avoid the numerical difficulties in computing the parametric images of the offset trimming loops on the offset surface;

3. for trimming loops, trimming curves are first propagated along the underlying surface normal with the given offset distance; then the corresponding surface curves on offset surface for

each of these 3D offset trimming curves are computed; finally, offset trimming loops are constructed using all the offset surface curves;

4. the offset trimmed NURBS surface is created using the extended offset surface and all the offset trimming loops.

This approach works under the assumption that the singularities should not occur after surface propagation, which means that the offset distance should be everywhere less than the local curvature radius.

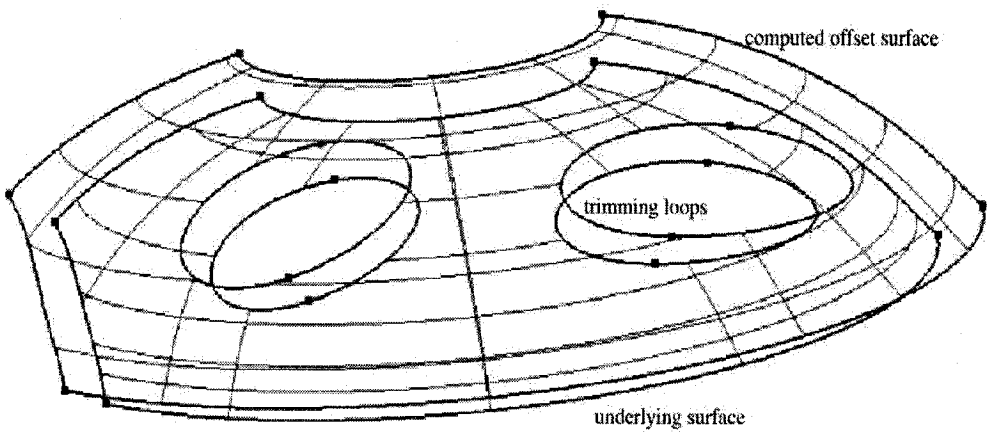


FIGURE 3.6 A trimmed NURBS surface (Kumar *et al.* (2003))

3.2.4 Marker Particle Method

The *marker particle method* is known under a variety of names, including marker particle techniques, string methods, nodal methods whose detailed descriptions can be found in Sethian (1982); Sethian (1985); Osher and Sethian (1988).

The difference between this method and the NURBS offset method lies in the way the normal directions are calculated. In the NURBS offset method, relations from differential geometry are used to represent normal vectors. While the marker particle method uses parameterized equations.

The unit front normal vector $\vec{n}$ in Eqn. (3.1) is parameterized into segments along the parameter direction s. Replacing $\vec{n}$ by finite difference approximations, offset curves or surfaces are generated by successively applying the resulting set of algebraic equations. Since this method captures a strong solution to the offset problem, self-intersection problems are unavoidable, and an additional scheme to detect and remove warped areas from the result is required. The front offsetting algorithm is given below:

1. *Front parameterization*

   The front is parameterized into segments along parameter s, thus the normal vectors can be replaced by the parameter derivatives.

$$\vec{n} = \vec{n}(x_s, y_s, z_s) \tag{3.8}$$

2. *Finite difference approximations*

   The numerical solution of Eqn. (3.8) is obtained by replacing the parameter derivatives at each grid point by central differences;

3. *Advancing-layer generation*

   Finally, these approximations of parameter derivatives are substituted into Eqn. (3.1), which yields a system of algebraic equations. The advancing-layer is generated by successively solving this transformed differential equation.

Helmsen (1994) has used de-looping procedures in an attempt to remove the wrapped areas (self-intersection area) directly from the result. While these procedures are sometimes manageable in two spatial dimensions, they become increasing intractable in three spatial dimensions (Sethian (1999)).

3.3 Indirect Offset Method

The indirect Offset Method models the front propagation problems into hyperbolic partial differential equations, in which both the initial front and propagated fronts are implicitly embedded into the equations. The numerical solution of the equations is the resulting front.

There are two ways of embedding the fronts into hyperbolic PDEs. The first is *the Level Set Method* which was developed by Sethian (1996). This method views the front propagation as an initial value problem where the resulting front is obtained by repeatedly iterating in time space. The other method is *the Eikonal Equation Based Method* which is promoted by this work. This method deals with the front propagation problem as an initial boundary problem. This section will discuss the methods used to solve these equations.

3.3.1 Level set method

The *Level set method* developed by Sethian and Osher (Sethian (1988)) models front propagation problems as a hyperbolic differential equation:

$$\phi_t + F|\nabla\phi| = 0 \tag{3.9}$$

where ϕ is the level set function, which represents the offset distance to the original curves or surfaces. F is the propagation speed function, and t is the time step. In this representation, the original front and propagated front are implicitly embedded into the Level Set model.

The main idea of the level set methodology is to embed the propagating interface as the zero level set of a higher dimensional function ϕ. The Level Set model links the evolution of the function ϕ to the propagation of the front itself through the time-dependent initial value problem given by Eqn. 3.9. At any time, the front is given by the zero level set of the time-dependent function ϕ. Or, in other words, the level set value of a particle on the front with path $\vec{x}(t)$ must always be zero at any time t.

The level set method employs a local Riemann scheme to solve the initial differential equation which uses values upwind of the direction of information propagation. A more comprehensive discussion of the Riemann scheme can be found in Engquist and Osher (1980). The numerical scheme for solving Eqn.(3.9) was given in Sethian (1996). The significant contribution of this scheme is that self-intersection problems are avoided in a natural manner. Kimmel applied level set equation to offset NURBS curves and surfaces in Kimmel and Bruckstein (1993).

Weaknesses of this method include: (1) The result of offsetting curves or surfaces are dependent on many other factors, i.e.: offset speed F, curvature k etc., which increase the complexity of the algorithm, and degrade the precision of offsetting curves or surfaces; (2) it requires re-initialization of the level sets during calculation to maintain a nice level set representation, which increases the cost of computing time.

3.3.2 The Eikonal equation based method

This research proposes to offset the front using the Eikonal equation. Many efforts were made towards solving the equation in the last two decades: upwinding schemes (Trier and Symes (1991), Vidale (1988)), dynamic programming sweeping methods (Schneider *et al* (1992)), Jacobi iterations (Rouy and Tourin (1992)), semi-Lagrangian schemes (Falcone and Ferretti (1994)), down-n-out approaches (Dellinger and Symes (1997), Kim and Cook (1999)), wavefront expanding methods (Qin *et al* (1992)), adaptive upwinding methods (Qian and Symes (2002)).

The fast sweeping methods (Boue and Dupuis (1999), Zhao *et al* (2000), Tsai *et al* (2003), Kao *et al* (2004), Zhang *et al* (2004), Zhao (2005)) and fast marching type methods (Tsitsiklis (1995), Helmsen *et al* (1996), Sethian (1996), Kimmel and Sethian (1998)) both are efficient methods designed to solve the nonlinear system directly using causality of the PDE. Only these two methods are discussed in this section.

The fast marching method has a computational complexity of $\Omega(M \log M)$, since a heap-sort is used at each step. M is the total number of mesh points. The fast sweeping method has a computational complexity of $\Omega(M)$ for Cartesian grids (Zhao (2005)). If the first order monotone scheme is used, both schemes will obtain the same accuracy, which is determined by the discretization size. In general only $O(0.5h)$ precision (Crandall and Lions (1984)) and $h \log h$ convergence can be expected (Zhao (2005)), with h the Cartesian mesh size.

The present work uses the fast sweeping method, because (1) it provides a better computational complexity ($\Omega(M)$) compared with the fast marching method ($\Omega(MlogM)$); (2) its implementation is straightforward and easier than the fast marching method, since its data structure is an array, while the fast marching methods uses heap data structure; and (3) the parallelized algorithm can

be directly applied to this scheme.

Fast Sweeping Scheme

The original fast sweeping method was inspired from the work of Boue and Dupuis (1999) using Gauss-Seidel iterations with alternate sweeping orderings.The crucial idea behind the fast sweeping method developed in Zhao (2005) is that all directions of characteristics can be divided into a finite number of groups; any characteristic can be decomposed into a finite number of pieces that belong to one of the above groups; there are systematic orderings that can follow the causality of each group of directions simultaneously.

On a rectangular grid there are natural orderings of all grid points. For example, in the two-dimensional case, there are four characteristic directions, up-right, up-left, down-right, and down-left, which yield four possible orderings to cover the entire computational domain.

Fast Marching Scheme

The basic idea of the fast marching method (Sethian (1996)) is summarized below. Initially, the closest points data in the zero band that surrounds the surface is computed. This closest point data in the zero band determines the closest points in the narrow band on the distance grid. Then this data on the narrow band is marched outward and inward to calculate the closest points in the rest of the distance grid.

Suppose there are M points in the distance grid. Each distance grid point is stored in a heap, and is removed from the narrow band once. The cost of adding and deleting elements from the narrow band is proportional to the logarithm of the number of points in the narrow band, thus the computational complexity of recomputing the distance is $\Omega(MlogM)$.

3.4 Conclusion

Overall, among all of these reviewed propagating methods, the advantage of the DOM methods is that the entire or partial original parameterization information is well preserved, but the self-intersections problem is addressed in an ad hoc way requiring heuristics for detection and removal. This does not lend itself to a fully automatic procedure.

The indirect offset methods prevail over direct offset methods, and bring a significant improvement and an elegant way in avoiding self-intersection problems. They intrinsically prevent self-intersections by constructing numerical solutions to the offset problem. In these methods, corners and cusps are naturally handled, and topological changes can take place in a straightforward and rigorous manner. Complex motion, particularly those that require surface diffusion, sensitive dependence on normal directions to the interface, and sophisticated breaking and merging, can be straightforwardly implemented, without user intervention (Malladi and Sethian (1996)).

For the above reasons, the present work applies the indirect method, but a simpler mathematical model - the Eikonal equation, for solving the offset problem. The fast sweeping method is used to solve the Eikonal equation. The derivation of the Eikonal equation and the details about the fast sweeping numerical scheme are presented in Chapter 4.

CHAPTER 4

THE SOLUTION OF THE EIKONAL EQUATION

Among the reviewed offset methods, the level set method prevails over other methods. It uses an upwind scheme to calculate the weak solution of the level set equation, which brings a significant improvement in the treatment of curves or surfaces local self-intersection problems during propagation.

However, the result of offsetting curves or surfaces are dependent on many factors, such as the calculation of offset speed F and curvature k, and re-initialization of the level set function during calculation are required to maintain an accurate level set representation. These requirements increase the computing time, and complicate the implementation. In this chapter, the use of the *Offset Distance Equation* instead of the level set PDE is proposed. This equation directly represents the connection between the offset distance ϕ and the space coordinates, thus simplifying the mathematical model. In addition, this results in a more computationally efficient algorithm

The fast sweeping method proposed by Zhao Zhao (2005) is directly used to solve the equation. The contributions of the presented work include: 1), extending the Eikonal equation to solve the front propagation problem; and 2) validation of continuous surface (Theorem 4.4 of Zhao (2005)) in 3-dimension using the fast sweeping method; 3) development of an initialization algorithm to compute initial ϕ values for continuous surface.

4.1 Introduction

The distinction between the proposed Offset Distance Equation and the level set method is that they deal with front propagation problems from different perspectives, that is, stationary and dynamic, respectively. In the Level set method, the front propagation problem is treated as an initial value problem in which initial values of the level set function ϕ are known in the computational domain. This ϕ function evolves in time in such a way that the zero level set of the function is always identified with the propagating interface. The zero level set needs to be extracted from the computational domain.

The present method casts the front propagation problem as the solution of a boundary value problem, in which the front to be offset is the zero level set and is viewed as the boundary condition. The iso-ϕ curves or surfaces which are equal to the given offset distance is the resulting offset and needs to be extracted from the computational domain.

The present method preserves all the advantages of the level set method, i.e., natural and accurate ways of tracking sharp corners and cusps, and handling subtle topological changes of merger and breakage, since both the level set method and the present method rely on viscosity solutions of the associated partial differential equations in order to guarantee that a unique, entropy-satisfying solution is obtained. In addition, the present method requires no time step, and hence its approximation is not subject to CFL stability conditions, unlike the level set method.

One of the limitations of the present method compared with the level set method is that it can solve only equal distanced propagation problems since there is no speed function embedded in the Eikonal equation.

As is also the case with the level set method, the parametric connection with the parent layer is not preserved. Offset surfaces are extracted from the computed ϕ field as iso-value surfaces. This results in a triangulated surface representation which has to be re-parameterized. Transforming this parameterization or connectivity information into offset surfaces is investigated in Chapter 5.

4.2 Derivation of the equation

Let Γ be the curve or surface to be propagated which can be closed or open; $\vec{P}$ be a point in the domain of interest Ω, $\Gamma \subset \Omega$; and ϕ be the smallest Euclidean distance between the point $\vec{P}$ and the curve(surface) Γ. To represent the distance function ϕ, a point $\vec{P_0}$ is defined which meets the following conditions:

1. $\vec{P_0}$ is located on Γ;
2. the Euclidean distance between $\vec{P}$ and $\vec{P_0}$ is the minimum Euclidean distance between $\vec{P}$ and Γ.

Then the ϕ function can be expressed as

$$\phi = |\vec{P} - \vec{P_0}|, \tag{4.1}$$

in which ϕ is 0 when $\vec{P}$ lies on Γ. In a 3D Cartesian frame of reference, let $\vec{P} = (x, y, z)$, $\vec{P_0} = (x_0, y_0, z_0)$, then Eqn. (4.1) can be written as

$$\phi(x,y,z) = \sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}. \tag{4.2}$$

The first derivatives of ϕ in the x, y and z directions are:

$$\begin{aligned}
\frac{\partial \phi}{\partial x} &= \frac{x - x_0}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}}, \\
\frac{\partial \phi}{\partial y} &= \frac{y - y_0}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}}, \\
\frac{\partial \phi}{\partial z} &= \frac{z - z_0}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}}.
\end{aligned} \tag{4.3}$$

A squared summation of the first derivatives (Eqn. (4.3)), yields the distance offset equation, which can be written in a more practical form:

$$\left(\frac{\partial \phi}{\partial x}\right)^2 + \left(\frac{\partial \phi}{\partial y}\right)^2 + \left(\frac{\partial \phi}{\partial z}\right)^2 = 1 \tag{4.4}$$

This equation can be cast into a more general and formal manner for any dimension or frame of reference as the boundary value problem:

$$\begin{cases} \nabla \phi \cdot \nabla \phi = 1 \\ \phi = 0 \quad if\ \vec{P} \in \Gamma \end{cases} \tag{4.5}$$

where ϕ is the minimum Euclidean distance between the front (Γ) to be propagated and P, an arbitrary point in the computational space Ω.

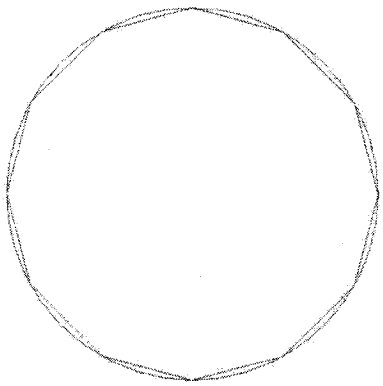
4.3 Discretization of the ϕ-field

The geometry information of the front to be propagated is given as line segments in 2D, or triangular faces in 3D. The representation of the original curves or surfaces does not impact on the method, as long as it provides intersection tests and closest point interrogation procedures. For instance, as illustrated in Fig. 4.1(a), a curve which is originally represented using NURBS, is first discretized into line segments before propagation.

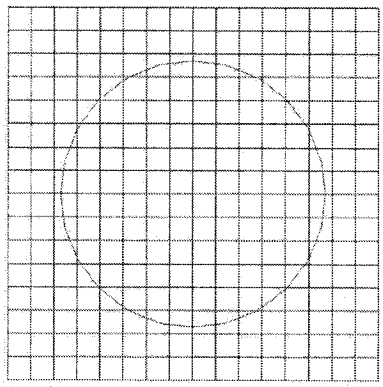
The second step is to define and discretize the domain Ω (see Fig. 4.1(b)). It is constructed by enlarging the bounding box of the original geometry by 5% to 40%. The domain Ω is discretized into an equally spaced Cartesian grid of square (cube) elements in $2D$ ($3D$), i.e. $h = \Delta x = \Delta y (= \Delta z)$. This is to simplify the procedure and is not a restriction of the algorithm.

The Cartesian grid nodes are classified into two types: *boundary-nodes*, and *non-boundary-nodes*. If Γ intersects an edge constructed between two Cartesian grid nodes, then all the nodes belonging to the cells sharing this edge are marked as boundary-nodes. In Fig. 4.1(c), suppose a boundary segment ($S_i S_{i+1}$) passes through an edge which consists of two Cartesian grid nodes (ab), then all the nodes belonging to the two neighboring cells are marked as boundary nodes. All the rest of the Cartesian nodes are called non-boundary nodes.

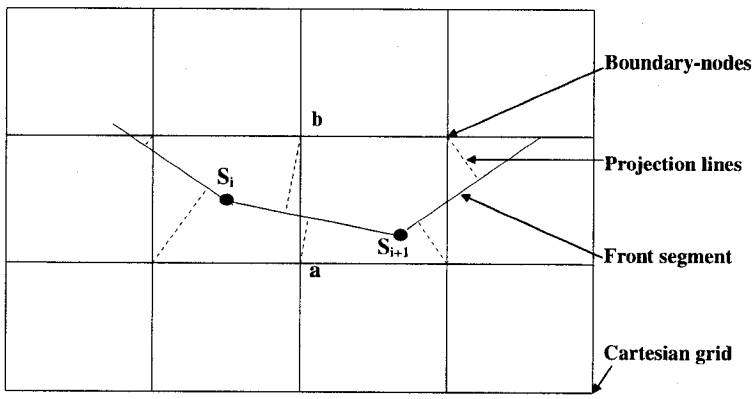
Boundary-nodes To enforce the boundary condition, $\phi = 0$ when $(x, y, z) \in \Gamma$, *exact distance (ϕ) values* are assigned to boundary-nodes. These are kept fixed during the sweeping calcula-



(a) Discretization of boundary Γ



(b) Construction and discretization of domain Ω



(c) Boundary-nodes definition

FIGURE 4.1 Problem discretization

tions. The exact value is calculated by projecting a boundary-node onto the front segments and the minimum length of this projection line is the value to be used for initializing the ϕ value of this boundary-node (see Fig. 4.1(c)). To accelerate boundary-node detection, the line segments in 2D (or triangles in 3D) are stored in an Alternating Digital Tree (ADT), as described by Bonet and Peraire (1991).

Non-boundary-nodes A large positive value is initially assigned to all non-boundary-nodes, which is updated subsequently by the sweeping algorithm.

4.4 Computation of ϕ

The following is an extended description of the *Fast Sweeping Method* algorithm presented by Zhao (2005), as applied in 3D in the present work.

A 3D Cartesian grid system is used to discretize Eqn. (4.4), with a uniform step size in the x, y and z directions, given as $\Delta x = \Delta y = \Delta z = h$. Distance function derivatives $\frac{\partial \phi}{\partial x}$, $\frac{\partial \phi}{\partial y}$ and $\frac{\partial \phi}{\partial z}$ are replaced by upwind differences,

$$\begin{aligned} \frac{\partial \phi}{\partial x} &= \frac{\phi_{i,j,k}^{t+1} - a}{h}, \\ \frac{\partial \phi}{\partial y} &= \frac{\phi_{i,j,k}^{t+1} - b}{h}, \\ \frac{\partial \phi}{\partial z} &= \frac{\phi_{i,j,k}^{t+1} - c}{h}. \end{aligned} \tag{4.6}$$

In the above equations, (i, j, k) are the indices for the Cartesian grid nodes in 3D, t is the sweep

counter, and a, b, c are defined as

$$\begin{aligned} a &= min(\phi^t_{i-1,j,k}, \phi^t_{i+1,j,k}) \quad i = 2, ..., I-1 \\ b &= min(\phi^t_{i,j-1,k}, \phi^t_{i,j+1,k}) \quad j = 2, ..., J-1 \\ c &= min(\phi^t_{i,j,k-1}, \phi^t_{i,j,k+1}) \quad k = 2, ..., K-1 \end{aligned} \tag{4.7}$$

In Eqn. (4.7), I, J and K are the total grid number in the x, y and z directions respectively. a, b and c are determined by their internal neighboring values when $i = 1$ or I, $j = 1$ or J and $k = 1$ or K. Replacing the partial derivatives into Eqn. (4.4) by their discrete approximations defined in Eqn. (4.6), yields:

$$\left[(\phi^{t+1}_{i,j,k} - a)^+\right]^2 + \left[(\phi^{t+1}_{i,j,k} - b)^+\right]^2 + \left[(\phi^{t+1}_{i,j,k} - c)^+\right]^2 = h^2 \tag{4.8}$$

where function $(m)^+$ is defined as:

$$(m)^+ = \begin{cases} m, \text{ when } m > 0 \\ 0, \text{ when } m \leq 0 \end{cases}.$$

Re-arranging a, b, and c so that $a < b < c$, the discrete solution $\phi^{t+1}_{i,j,k}$ to Eqn. (4.8) can be obtained by the following algorithm:

let $\phi_{temp} = a + h$

1. if, $\phi_{temp} \leq b$, then $\phi^{t+1}_{i,j,k} = \phi_{temp}$, and it is the solution to Eqn. (4.8);
2. else, let $\phi_{temp} = \frac{(a+b)+\sqrt{2h^2-(a-b)^2}}{2}$

(a) if, $\phi_{temp} \leq c$, then $\phi_{i,j,k}^{t+1} = \phi_{temp}$, and it is the solution to Eqn. (4.8);

(b) else, $\phi_{i,j,k}^{t+1} = \frac{(a+b+c)+\sqrt{3h^2+2(ab+bc+ac-a^2-b^2-c^2)}}{3}$, and it is the solution to Eqn. (4.8).

The new value $\phi_{i,j,k}^{new}$ at node (i, j, k) is chosen as: $\phi_{i,j,k}^{new} = min(\phi_{i,j,k}^{t}, \phi_{i,j,k}^{t+1})$.

The old value $\phi_{i,j,k}^{t}$ is always replaced by $\phi_{i,j,k}^{new}$ in the sweeping calculation process.

To enforce the boundary condition, exact values at Cartesian grid points on or near Γ are assigned by projecting them to the front as illustrated in Fig. 4.2. Then the fast sweeping algorithm is applied to calculate the ϕ field for the rest of grid points in the Cartesian domain.

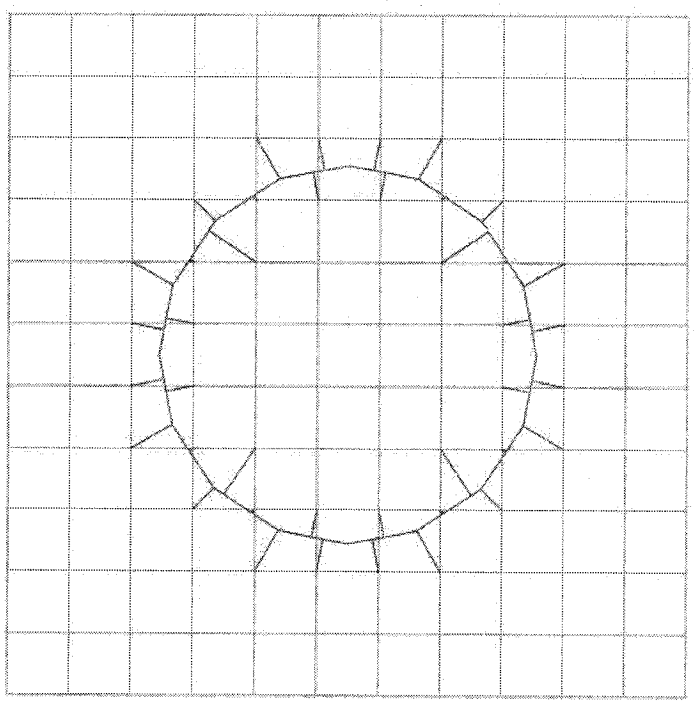


FIGURE 4.2 Initializing exact ϕ for all near boundary nodes

Algorithm 1 updating ϕ (one sweep example)

for $k = 1, K$ **do**
  for $j = 1, J$ **do**
    for $i = 1, I$ **do**
      1) calculate a, b, and c
      2) calculate $\phi_{i,j,k}^{t+1}$
      3) calculate $\phi_{i,j,k}^{new}$
        $\phi_{i,j,k}^{new} = min(\phi_{i,j,k}^{t}, \phi_{i,j,k}^{t+1})$
    end for
  end for
end for

Gauss-Seidel iterations are used to update ϕ's at non-boundary-nodes. Algorithm 1 gives an example for one sweep in 3D.

One-sided difference schemes are used at the boundaries of Ω for the values of a, b, and c in the updating process. i.e. $a = \phi_{2,j,k}^{t}$ when $i = 1$; and $a = \phi_{I-1,j,k}^{t}$ when $i = I$. The same rule applies for b and c values at boundaries of Ω. The one-sided difference schemes at the boundaries of Ω enforces the propagation of information from inside the domain Ω to the boundary, since the data set Γ is contained in Ω. For example, at boundary $x_{1,j,k}$, the following equation is used:

$$(\phi_{1,j,k}^{t+1} - \phi_{2,j,k}^{t})^2 + (\phi_{1,j,k}^{t+1} - b)^2 + (\phi_{1,j,k}^{t+1} - c)^2 = h^2 \tag{4.9}$$

4.5 Validation of Accuracy

4.5.1 Norms (L_2 and L_∞) definitions

In the accuracy validations, we use L_2 and L_∞ norms to analysis the results, which definitions are given below:

L_2 represents the local norm and it is defined as

$$L_2 = ((\sum_{i=0}^{n}(\phi_{i_{exact}} - \phi_{i_{appro}})^2)/n)^{1/2} \tag{4.10}$$

L_∞ represents the global norm and it is defined as

$$L_\infty = (\sum_{i=0}^{n}(\phi_{i_{exact}} - \phi_{i_{appro}}))/n \tag{4.11}$$

where $\phi_{i_{exact}}$ is the exact minimum Euclidean distance to the front from point P_i ($P_i \in \Omega$, Ω is the computational domain), $\phi_{i_{approx}}$ is the minimum Euclidean distance calculated by the fast sweeping scheme, n is the total number of the Cartesian grid points.

4.5.2 Theorems

We will use Theorems 4.1, 4.3 and 4.4 for accuracy validation purpose. These theorems are proved in Zhao's work and are used directly in this research.

In Theorem 4.1, Zhao (2005) states that for a single data point, the numerical solution of the fast

sweeping method converges after 2^n sweeps in R^n, and the error of the numerical solution by the fast sweeping method is bounded by $\Omega(|hlog(h)|)$, where n is the spatial dimension, and h is the Cartesian grid size.

In Theorem 4.3, Zhao (2005) states that for an arbitrary set of discrete points, the error of the numerical solution by the fast sweeping method is bounded by $\Omega(|hlog(h)|)$ after 2^n sweeps in R^n, where n is the spatial dimension, and h is the Cartesian grid size.

In Theorem 4.4, he states that for smooth curves or surfaces, the error of the numerical solution by the fast sweeping method is bounded by $\Omega(hlog(\frac{1}{h}))$ after 2^n sweeps in R^n, where n is the spatial dimension, and h is the Cartesian grid size.

4.5.3 Test cases

The presented results show that all computations are carried out using four sweeps in two dimensional domain and eight sweeps in three dimensional domain. Convergence is not influenced by grid size, but the spatial dimension only.

2D test

The fast sweeping scheme has been implemented in 2D and validated with example 2 in Zhao's work. In this test, the domain is a unit square, a single data point is located at $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{3}})$. Table 4.1 shows the errors measured in different norms with different grid sizes. For one single point, the fast sweeping method converges exactly in four sweeps in two dimensions, which is coincident with Theorem 4.1 in Zhao (2005). Fig 4.3 is the graphical illustration of the comparison of the norms for this test. From this figure, we can see that L_2 and L_∞ is quasi-linearly convergent with

h	0.02	0.01	0.005	0.0025
$\|h * log(h)\|$	0.0340	0.02	0.0115	0.0065
L_2	0.02885	0.01386	0.00554	0.00307
L_∞	0.04981	0.02440	0.01097	0.00672

TABLE 4.1 Norms Comparison: a single data point

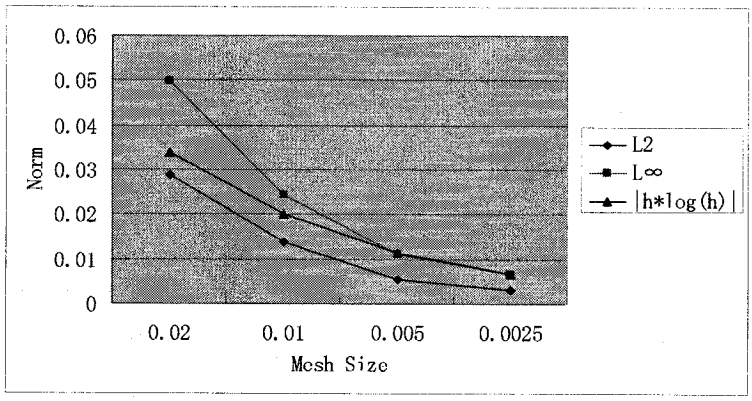


FIGURE 4.3 Norms Comparison: a single data point

mesh size, and L_2 is bounded by the maximum error as stated in Theorem 4.1.

3D tests (sphere and cube)

In addition, two simple 3D analytical test cases have been used to determine the accuracy of ϕ for the proposed algorithm. These test cases are a *sphere*, centered at $(0, 0, 0)$ with radius $r = 0.15$, with a Cartesian frame of 0.5x0.5x0.5, and a *cube* centered at $(0, 0, 0)$ with size of 0.3x0.3x0.3, and a Cartesian frame of 0.5x0.5x0.5. In these cases, the exact solutions are known, so that the convergence behavior of the method can be easily evaluated.

From Theorem 4.3 and 4.4, we can see that the error of the fast sweeping scheme is dependent on the Cartesian mesh size. For this reason, the error of the computed solution is only analyzed

h	**0.05**	**0.025**	**0.0125**	**0.00625**
$\|h * log(h)\|$	0.065	0.04	0.024	0.014
L_2	0.0100132	0.0056586	0.00299202	0.00154483
L_∞	0.0397926	0.0257559	0.0160101	0.00965976

TABLE 4.2 Norms Comparison: Sphere case

as a function of the Cartesian mesh size. For the sphere, the original front is specified as isolated points, and the initial ϕ values at all the boundary-nodes are initialized using the exact distances to the sphere surface. For the cube, the discretization for the geometric front is specified, and the behavior of the two norms (L_2 and L_∞) is studied as a function of the mesh size.

Table 4.2 presents the comparison of the two norms for meshes varying from 0.05 to 0.00625 for the sphere. In this case, the sphere surface, Γ, is viewed as an arbitrary set of discrete points, thus the predicted bounding error behavior is $|h\log(h)|$ (see *Theorem 4.3* of Zhao (2005)). Table 4.3 presents the norms comparison for the cube, for mesh size h varying from 0.02 to 0.0025. According to Zhao (2005) (see *Theorem 4.4*), the expected bounding error behavior in this case is $h\log(1/h)$, since the cube surface, Γ, is viewed as a smooth surface. Table 4.2 and 4.3 show that the results converged after eight sweeps in sphere and cube cases.

Figs. 4.4 and 4.5 are the graphical illustrations of the comparisons of the norms for the sphere and cube respectively. From these figures it can be seen that L_2 and L_∞ are quasi-linearly convergent with mesh size, and bounded by the maximum error predicted by Zhao (2005). These results confirm the expected convergence of the scheme using eight sweeps in 3D.

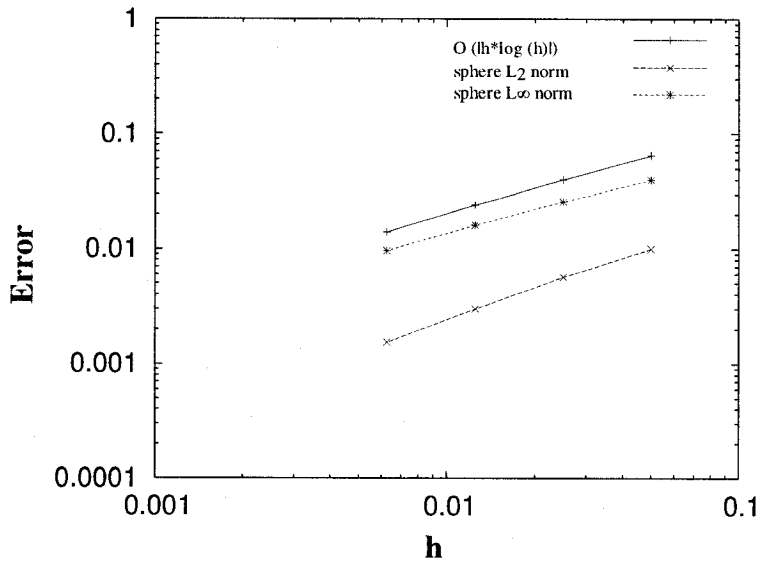


FIGURE 4.4 Norms Comparison: Sphere case

h	**0.02**	**0.01**	**0.005**	**0.0025**
$h * log(1/h)$	0.034	0.02	0.0115	0.0065
L_2	0.00353006	0.00188081	0.00106662	0.000606601
L_∞	0.0224117	0.0149685	0.00950699	0.00567656

TABLE 4.3 Norms Comparison: Cube case

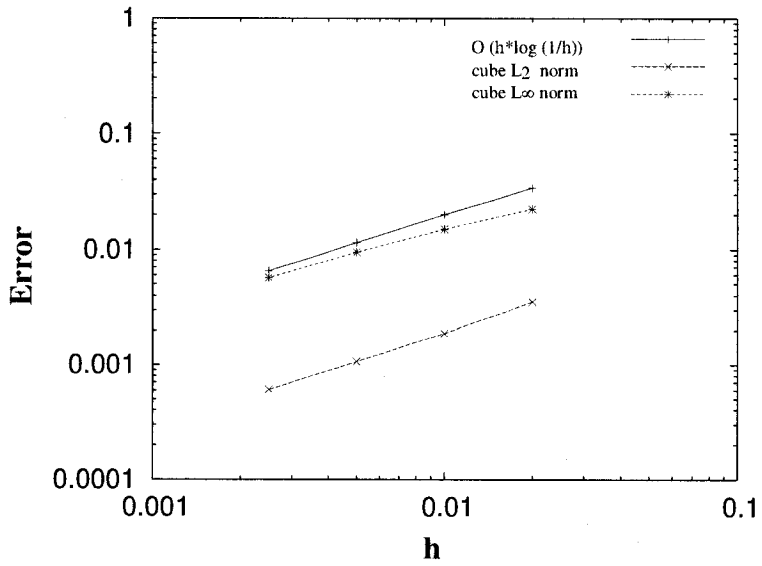


FIGURE 4.5 Norms Comparison: Cube case

4.6 Applications

The surface offsetting methodology described in the previous sections is now applied to the problem of near body domain decomposition through the generation of thin skins near solid boundaries in several different configurations. This is achieved by capturing iso-ϕ curves in 2D or surfaces in 3D obtained by interpolating the distance field values at the Cartesian grid nodes. Since all the interpolating points are isolated points, the resulting surfaces are in the form of triangulated surfaces in 3D.

4.6.1 Validation of topology

The purpose of these test cases is to illustrate how the surfaces are naturally broken at the cusps without user intervention. Fig 4.6 shows the inward propagation results of two different geometric

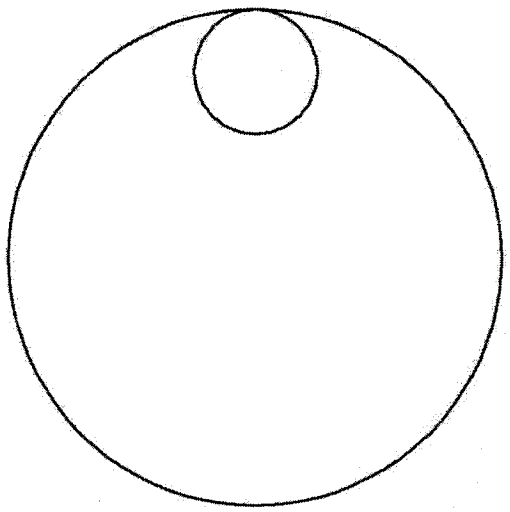
configurations. Fig. 4.6(a) defines two circles, the inner circle being tangent to the outer one. Fig. 4.6(c) consists of one closed line with a self-intersection loop at the top of the domain.

Fig. 4.6(b) and 4.6(d) show that both configurations behave identically during propagation. In both cases, the size for the Cartesian grid is 0.01, and the propagation distance is 0.02. From these results, we can see that when the original geometries are inwardly propagated a topological change occurs and the original curves are split into two curves.

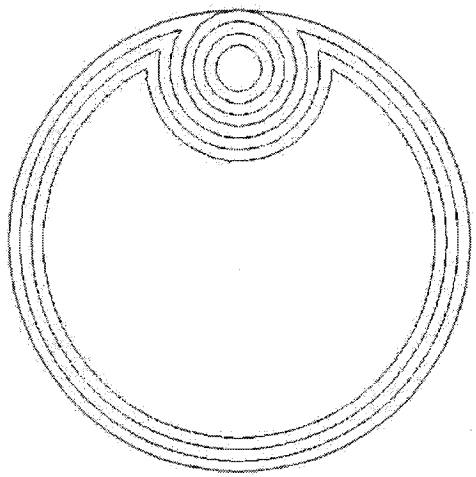
4.6.2 Turbine cascade

Propagation of the two-dimensional turbine cascade model shown in Fig. 4.7(a) demonstrates how the scheme avoids global warping problems. The two closed curves represent the stay vane and the wicket gate blades of a turbine distributor which are the solid boundaries in direct contact with the fluid. The two outside curves are periodic lines, which are artificial interfaces added for studying the flow phenomena in multiply-bladed turbo-machines. The objective is to generate thin layers around the two blades by outwardly offsetting their fronts. In this case, the two closed curves forming the fronts may clash with each other when the propagating distance is greater than half of the minimum distance between the two blades. In addition, for a given propagating distance, there are two iso-curves with the same ϕ value for each blade, one inside, and the other outside the blades. To correctly identify an inward or outward propagation in a closed domain, the sign of ϕ at each Cartesian grid is defined as follows:

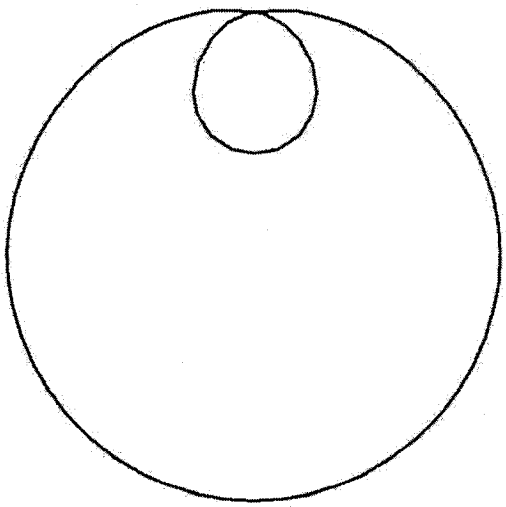
- $\phi > 0$ when a Cartesian grid is outside the blades;
- $\phi = 0$ when a Cartesian grid is on the blades;



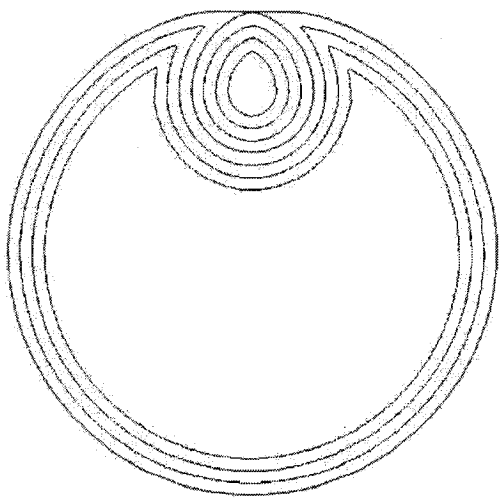
(a) Two tangential circles



(b) Inward propagation result

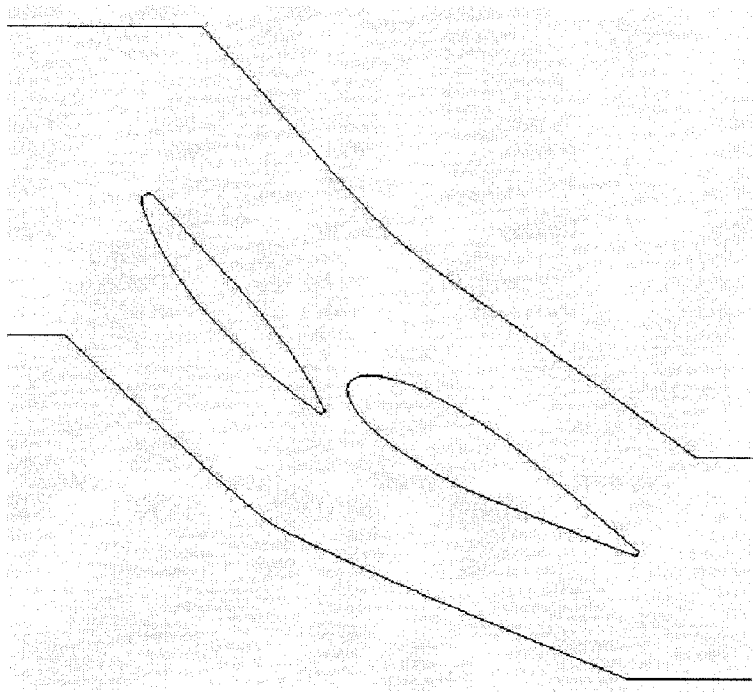


(c) A self-intersection loop

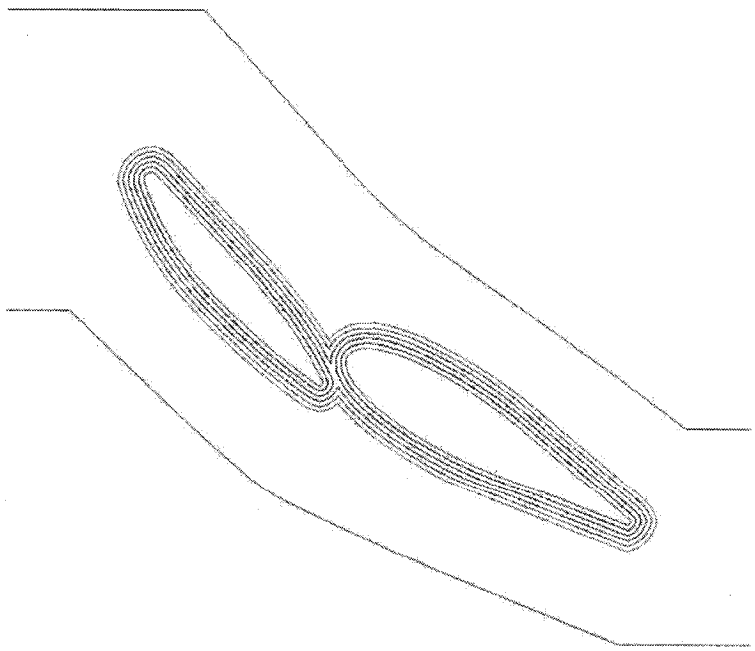


(d) Inward propagation result

FIGURE 4.6 Results



(a) Geometry definition



(b) Sample output

FIGURE 4.7 Turbine cascade

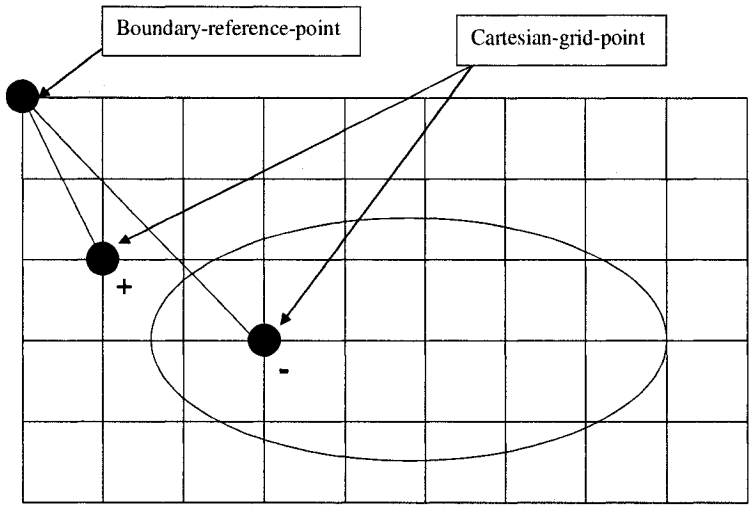


FIGURE 4.8 ϕ sign detection

- $\phi < 0$ when a Cartesian grid is inside the blades;

Figure 4.7(b) illustrates a sample output in which the fronts are outwardly advanced by the same value at each propagation step. It is noted that when the propagation distance is greater than a certain value, the two fronts do not conflict with each other, illustrating that global self-intersection is naturally avoided by the scheme itself. However, the trade-off is that topological information from previous levels is changed when the two propagated curves merge into a single front.

Algorithm to determine the sign of the ϕ function

The procedure to determine the sign of the ϕ function is shown in Fig. 4.8:

1. construct a line from a Cartesian grid point to the boundary reference point, which can be any point located at the boundary of Cartesian frame.
2. detect the number of intersections between this line and the geometric boundaries. If the

number of intersections is an even number, then the sign at this grid point is positive; otherwise it is negative.

4.6.3 Heat exchanger

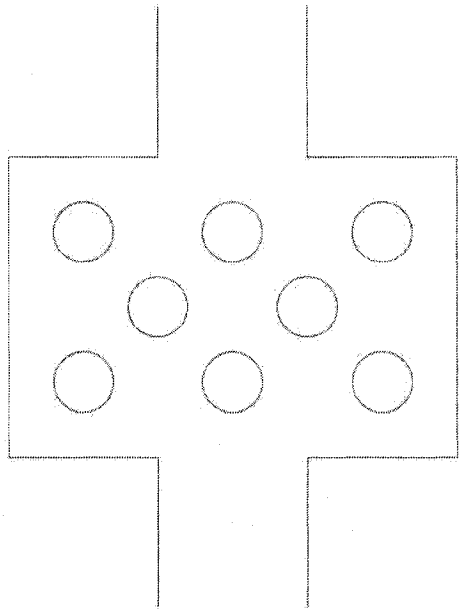
To illustrate the ability of the scheme to handle the propagation in a multi-connected open domain, with sharp corners, a heat exchanger configuration was used (Fig. 4.9) including circles in the middle representing the pipes, surrounded by a shell. Both the internal shell of the heat exchanger and the outside walls of the pipes are in direct contact with the fluid. For the purpose of studying the internal flow inside this configuration, all the pipes are outwardly propagated, and the shell of the exchanger is inwardly propagated.

Figure 4.9(a) is the geometry of the heat exchanger without defining inlet and outlet boundaries. These are introduced as artificial permeable interfaces in Fig. 4.9(b) and make the exchanger a closed domain.

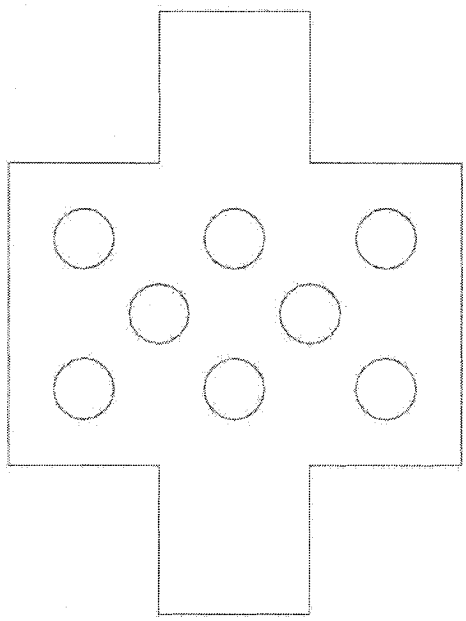
As shown in Fig. 4.10 the front is not differentiable as there is a sharp corner. The results show both convex and concave propagation behaviors of this discontinuous front as the shell moves along its inward and outward normal directions. After four sweeps, the distance between each propagation is 0.2. Fig. 4.10(a) and Fig. 4.10(b) show the propagation of this sharp front for two different meshes, i.e. 61x81 and 241x321, respectively.

From the results of Fig. 4.10, it can be seen that:

- The present method does not require the initial front to be differentiable, i.e. the method can handle a front with discontinuous points.

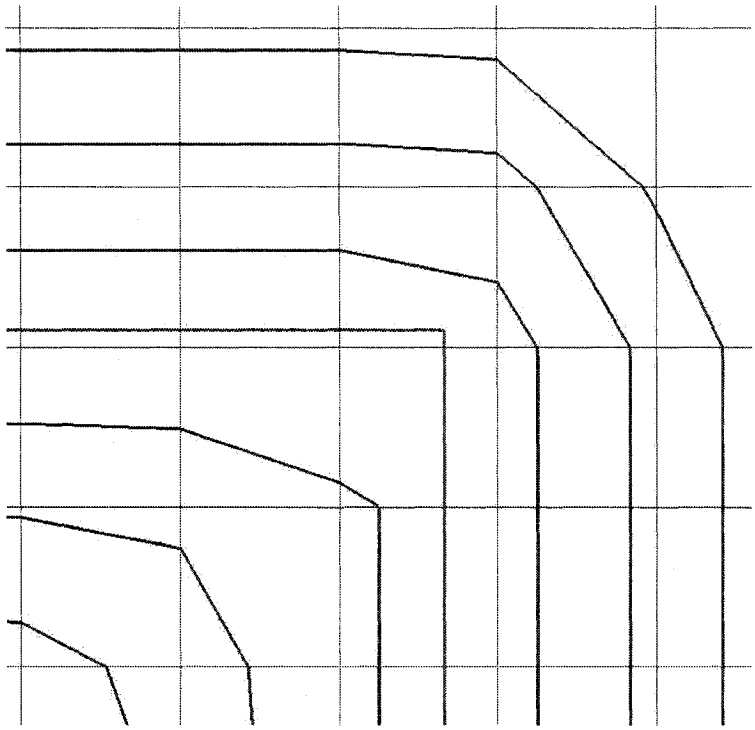


(a) Without inlet and outlet

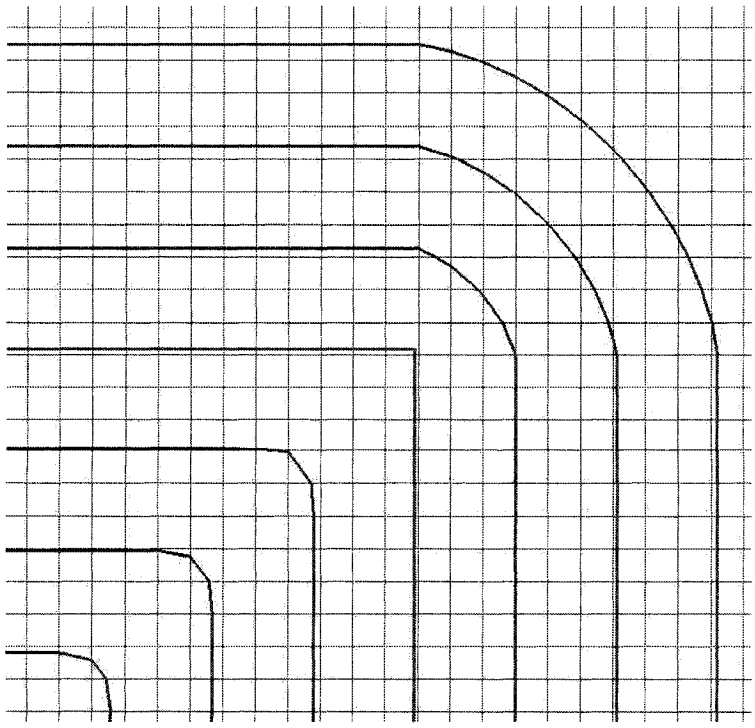


(b) With inlet and outlet

FIGURE 4.9 Heat exchanger configuration



(a) Mesh size = 61x81



(b) Mesh size = 241x321

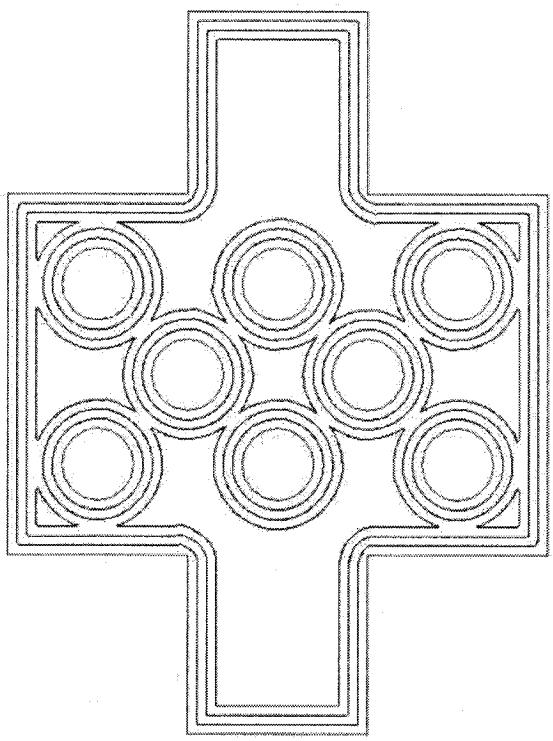
FIGURE 4.10 Corner propagation

- Local self-intersection problems are naturally avoided by the numerical scheme itself without adding any extra self-intersection detection and removing algorithm.
- If the direction which blunts the sharp corner is defined as the outward propagation, and the direction which sharpens the sharp corner is defined as the inward propagation, then the present method can deal with both convex and concave geometries without special treatment. The shell can be viewed as a convex shape when the shell is inwardly propagated, or a concave shape when it is outwardly propagated.
- The behavior of corner propagation can be controlled by varying the mesh size which directly influences the propagation behavior. As the mesh is refined, a sharp corner will be uniformly blunted or rapidly sharpened.

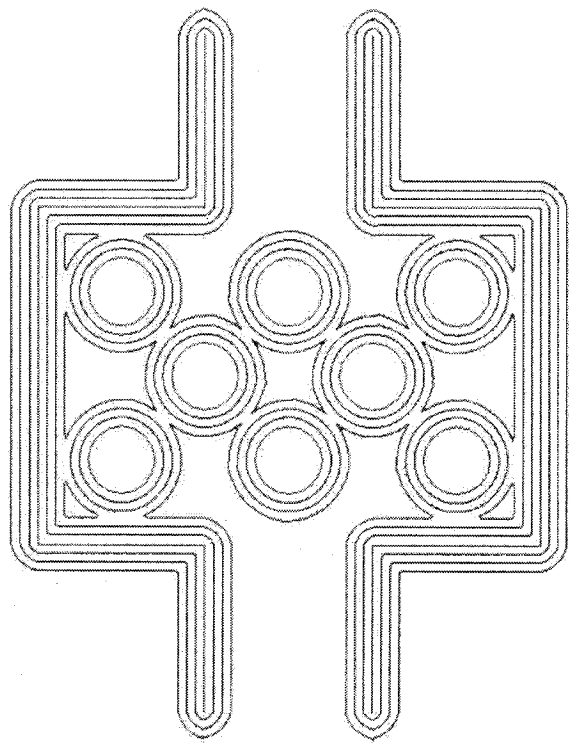
Figure 4.11 illustrates three distinct configurations corresponding to different boundary conditions, the detail about each boundary configuration is discussed below. The results are all obtained using the same mesh definition and a propagation distances between iso-curves of 0.2.

Figure 4.11(a) is the propagation of the front defined in Fig. 4.9(b), in which the inlet, the outlet and the shell of the heat exchanger are treated as one closed domain. The sign of ϕ is defined as follows:

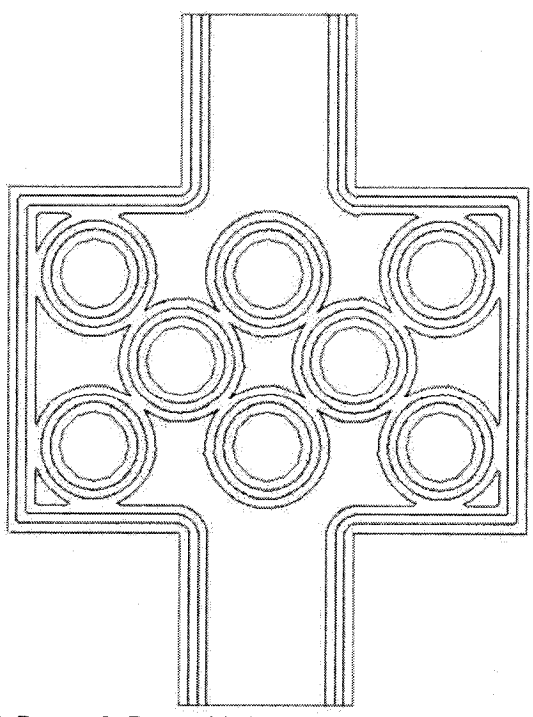
- $\phi > 0$ when a grid node is outside the shell (including inlet/outlet) of the exchanger; or when a Cartesian grid is inside the holes (the pipes);
- $\phi = 0$ when a grid node is on Γ;
- $\phi < 0$ when a grid node is inside the shell and outside the holes (the pipes).



(a) Pattern 1- Solid inlet/outlet boundary conditions



(b) Pattern 2- No inlet/outlet boundary conditions



(c) Pattern 3- Permeable inlet/outlet boundary conditions

FIGURE 4.11 Boundary conditions

Fig. 4.11(b) is the propagation pattern for the configuration of Fig. 4.9(a). In this case the shell of the heat exchanger is an open boundary, and to correctly capture the propagated curves, the sign of ϕ is set as follows:

- $\phi = 0$ when a grid node is on Γ;
- $\phi < 0$ when a grid node is inside the pipes;
- $\phi > 0$ elsewhere.

The appropriate configuration depends on the nature of the physical model. For instance, in through flow configurations neither Fig. 4.11(a) nor 4.11(b) are correct propagation patterns for studying the viscous flow in such a geometry. To obtain a correct propagation, a new type of boundary is required. These are inlet/outlet surfaces and are not solid but rather porous and open. These special boundary conditions must be applied to boundaries which do not propagate. It is a restriction to the propagation treatment, which confines the propagation path of extreme points to a preset trajectory. In Fig. 4.11(c), extreme points are the intersection points between the shell and the inlet or outlet surfaces. In general, all extreme points are marked in the post processing step, and they should meet two constraints: (1) their ϕ values are set equal to the propagated distance, and (2) their propagation paths lie on special boundaries.

To get Fig. 4.11(c), we first capture Fig. 4.11(b), and then add the inlet and outlet as special boundary conditions into Fig. 4.11(b) (see Fig. 4.12(a)). The porous inlet and outlet surfaces are then used to trim the extra part located outside of the shell (see Fig. 4.12(b)). Figure 4.12(c) shows the final inward propagation result.

4.6.4 Draft tube

To illustrate the importance of correctly treating of boundary conditions in real applications, the propagation method is applied to an industrial configuration in hydraulic turbo-machinery. The geometry consists of an external shell, the draft tube, with two internal obstructions, the piers and an extension. As shown in Fig. 4.13, it is essentially a duct or a cylindrical surface, bent into an elbow where the cross-section makes a transition from a circle to a square. To study the internal flow for this draft tube, the domain is decomposed into several partitions consisting of a skin in the vicinity of the solid walls and an internal core volume. Such multiple blocks are necessary to generate high aspect ratio elements in the grid generation process.

From the point of view of surface propagation, the geometry in this example is rather complex: it is a multi-connected domain, because there are two holes (piers) inside the domain; it includes both convex and concave shapes in its outside wall; the input geometry is not smooth, it has sharp corners; boundary conditions require special treatment for a correct propagation at the inlet and outlet surfaces. These through-flow (porous) surfaces exactly seal the entrance and exit part of this draft tube, and make it a closed domain. This leads to two distinct propagation problems depending on the way propagation is treated. In the first instance (see Fig. 4.14), the inlet and outlet surfaces are propagated as part of the draft tube surface. The draft tube's external shell is entirely shrunk along its internal normal direction. For fluid mechanics analysis, it is not the expected propagation result, because the extreme points located at the rim of the entrance and exit surfaces are not propagated correctly.

In the second instance (see Fig. 4.15), the draft tube surface is propagated distinctly from the inlet and outlet surfaces. These are not propagated and thus make the draft tube an open domain. In this test, the wall of the draft tube is dealt with as an open domain and the propagated surface looks

like a hollowed shell which wraps around the original front. To get the correct propagation result, an algorithm is developed to remove the extra part which is located outside of the inlet, outlet and the wall of the draft tube. Fig. 4.16 presents the correct output from two different points of view after removing the extra part. In this case, the wall of the draft tube is correctly propagated in its inward direction. The inlet and outlet surfaces work as special boundary conditions, and restrict the extreme points located at the entrance and exit part to be propagated correctly.

4.7 Discussion

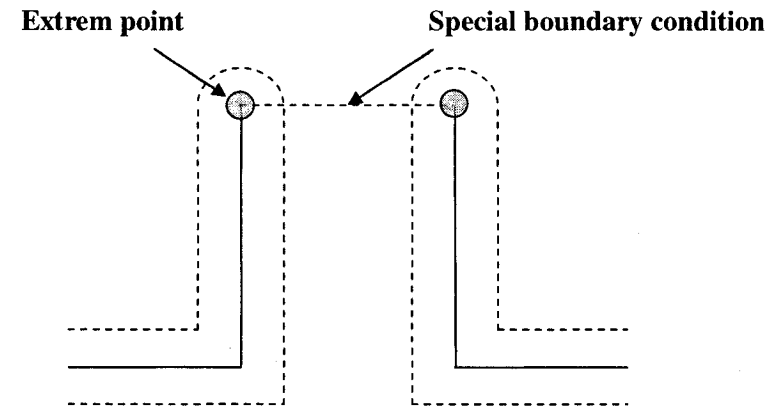
The front offset method discussed in this chapter casts the front propagation problem as the solution of a boundary value problem, in which the front to be offset is the zero level set and is viewed as the boundary condition. Both the original front and the resulting propagation are implicitly embedded into the equation. The iso-ϕ curves or surfaces which are equal to the given offset distance is the resulting offset. The offset front needs to be extracted from the computational domain.

The present method preserves all the advantages of the level set method, i.e., natural and accurate ways of tracking sharp corners and cusps, and handling subtle topological changes of merger and breakage, since both the level set method and the present method rely on viscosity solutions of the associated partial differential equations in order to guarantee that a unique, entropy-satisfying solution is obtained. In addition, the present method requires no time step, and hence its approximation is not subject to CFL stability conditions, unlike the level set method.

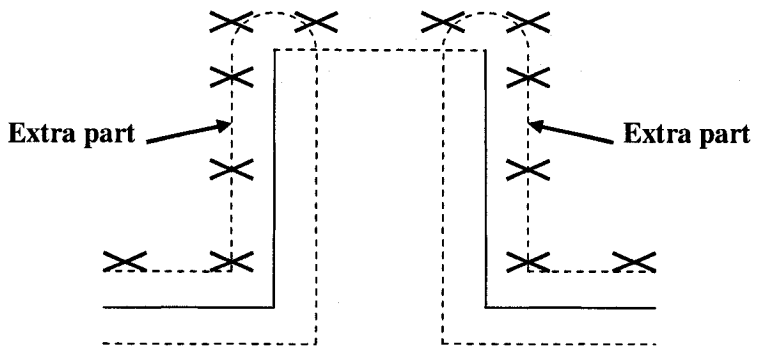
The presented work uses the fast sweeping method, because (1) it provides higher computational efficiency ($\Omega(M)$) compared with the fast marching method ($\Omega(MlogM)$); (2) its implementation is straightforward and easier than the fast marching method, since its data structure is an array,

while the fast marching methods uses heap data structure; and (3) the parallelized algorithm can be directly applied to this scheme.

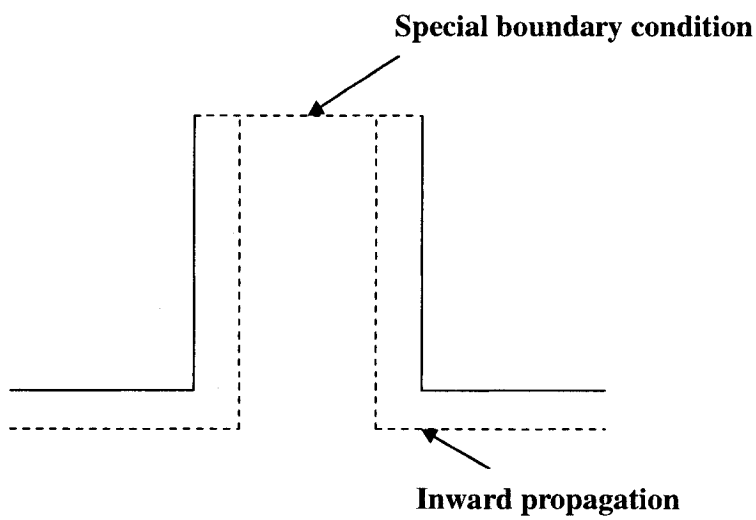
The ϕ field computed by the present method can be used to calculate the normal directions for propagating front, and the extended applications are presented in the next three chapters (Chapter 5 discusses the offset surface construction; Chapter 6 discusses the boundary layer mesh generation; and Chapter 7 discusses a fluid flow simulation).



(a) Special boundary condition added at outlet



(b) Extra part to be removed



(c) Inward propagation result

FIGURE 4.12 Special Boundary Condition

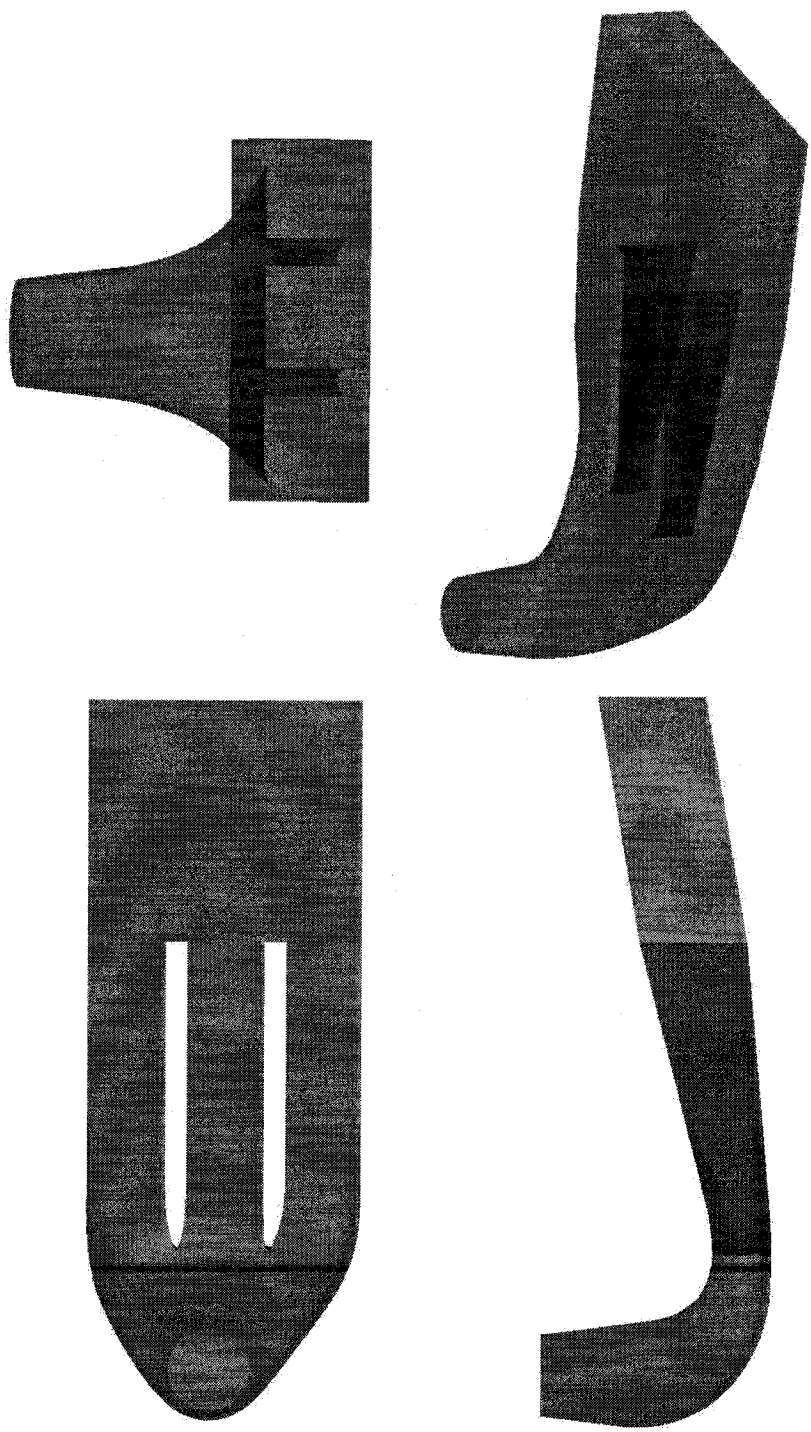


FIGURE 4.13 Draft tube (geometry)

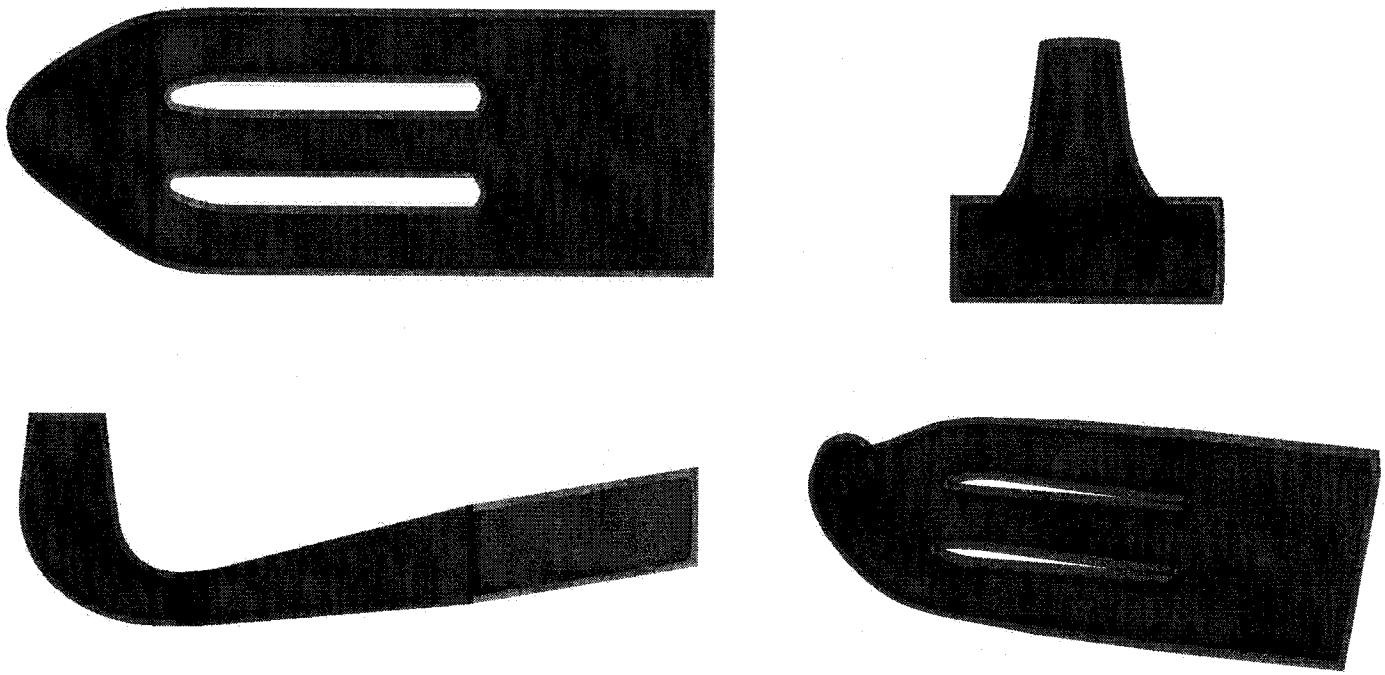


FIGURE 4.14 Offset with inlet/outlet

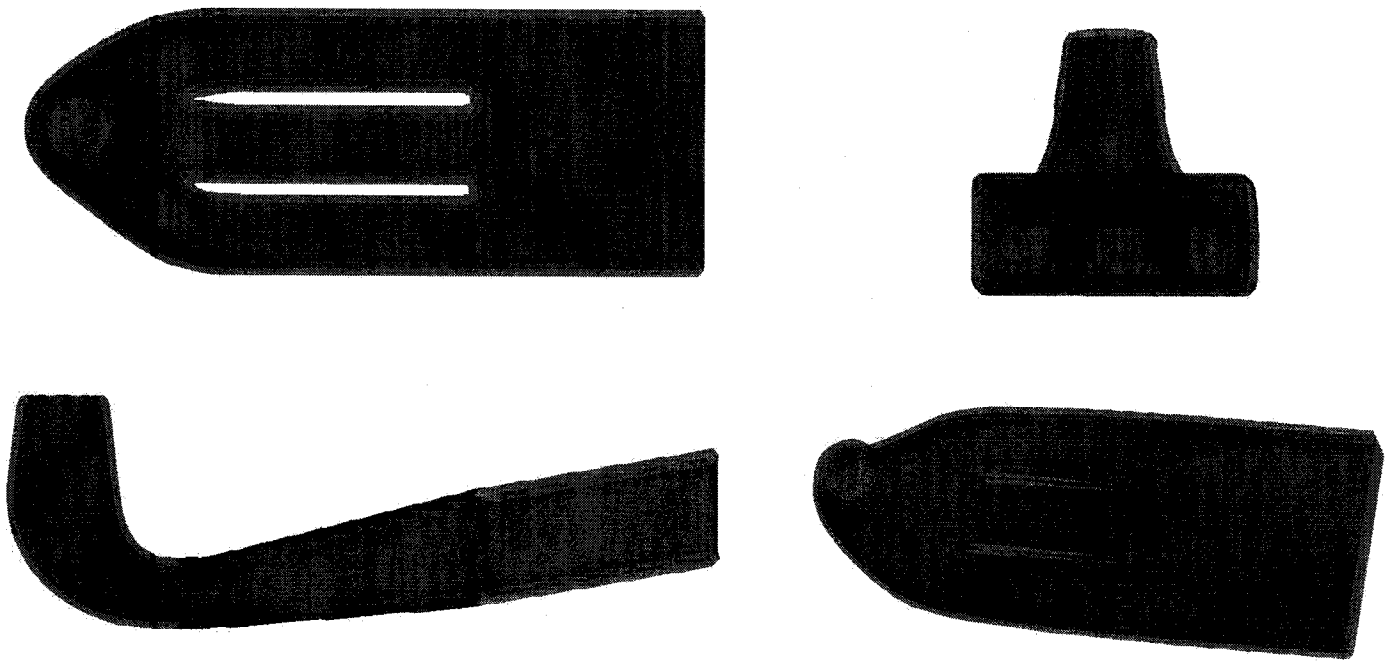


FIGURE 4.15 Offset before removing extra part (without inlet/outlet)

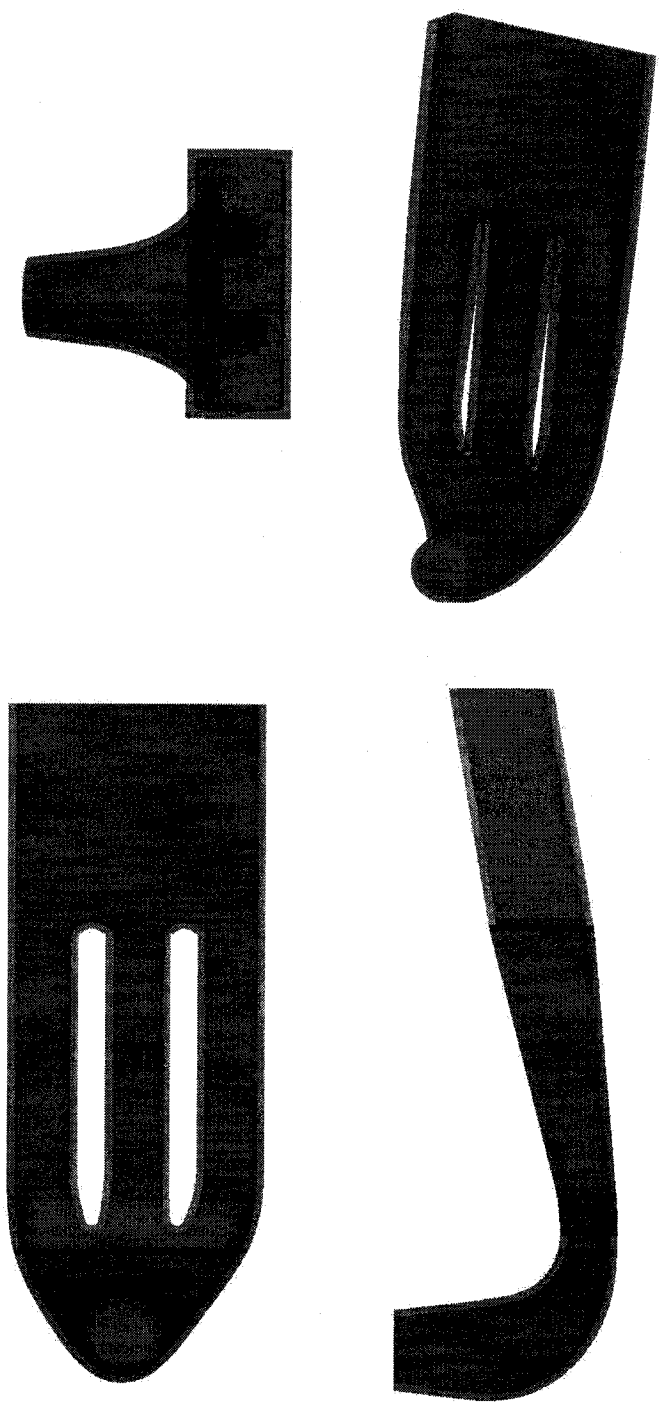


FIGURE 4.16 Offset after removing extra part (without inlet/outlet)

CHAPTER 5

APPLICATION 1 - APPROXIMATE OFFSET SURFACE CONSTRUCTION

As discussed in Chapter 2, the front propagation methodology based on the solution of a distance function does not preserve the parametric connection with the parent layer. In this chapter a new approach is proposed to enforce parametric consistency between Γ, the original surface, and Γ_p, its offset. This is achieved by a combination of the traditional direct offset techniques and the present Eikonal-based approach.

The main difference between the current method and traditional DOM methods lies in the manner in which the local normal direction is computed. In DOM, the local normal directions are computed using geometric information, such as the positions of neighboring points located on Γ. The central idea of the method is thus to propagate each discretized point of the original geometric front Γ along its normal direction in ϕ space rather than geometric space. This results in a methodology that in addition to resolving the self intersection problems, will also maintain the parametric relationship between Γ and Γ_p in most circumstances.

In this work, only discrete surfaces are considered. The input geometric description is represented using tesselated surfaces (i.e., triangulated cells or quadrilateral cells), the propagation methodology is discussed in Section 5.1 and its validation is illustrated in Section 5.4. This is the main topic of this chapter;

5.1 Methodology

The original front can be represented either in NURBS form or discretized form (lines in 2D or triangles in 3D). This section discusses the propagation of discretized curves/surfaces, described using NURBS which can be descritized with arbitrary precision.

The given geometry Γ is discretized by line segments (2D) or triangulated surface (3D), and is embedded into a domain Ω large enough to cover Γ. The offset is achieved by propagating all the points located on Γ along their local normal direction using the following equation:

$$\vec{x}^{n+1} = \vec{x}^{n} + \vec{F}dt \tag{5.1}$$

where F is the unit normal propagation velocity.

We are now facing the problem of representing the relationship between the offset distance ϕ and the space coordinates. Using the discretization procedure described in Section 4.3, and the algorithm of Section 4.4, the minimum Euclidean distance ϕ is computed at each Cartesian grid node. This field is then used to evaluate the local unit normal propagation velocity $\nabla\phi/|\nabla\phi|$ for each node on Γ.

After propagation, the final $\vec{x}^{n+1}$ are sequentially connected to construct Γ_p according to their original connectivities. Thus parametric consistency between Γ, the original surface, and Γ_p, its offset, is preserved, and all the points on Γ and Γ_p share a one-to-one parametric mapping. However, when two fronts collide, the parametric information can not be maintained, such as in the case illustrated in Fig. 5.1.

There are three essential steps in this method :

1. The domain Ω is discretized using a uniform Cartesian grid;
2. The ϕ field is computed for each Cartesian grid node using the fast sweeping algorithm;
3. Each node located on Γ is offset along its local normal direction according to the given distance.

For each propagated point, ϕ values are obtained using the ϕ values at Cartesian grid points by interpolation. From these, stopping and collision detection criteria are evaluated.

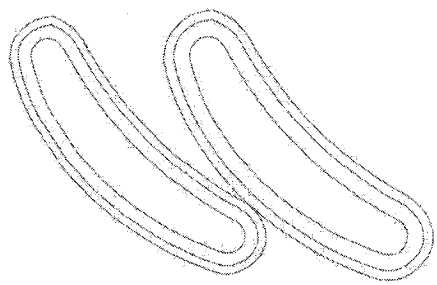


FIGURE 5.1 Collision detection criteria

Two stopping criteria are applied to each new propagated point $\vec{x}^{n+1}$:

- If ϕ is equal to or greater than the specified propagation distance, i.e. $\phi \geq d$;
- If the fronts to be propagated are too close, collisions between two offset fronts may occur for a given propagation distance as shown in Fig. 5.1. In this case, when ϕ starts to decrease, the condition $\phi_{n+1} - \phi_n < 0$ is used to stop propagation.

5.2 Propagation of the Parameterization

The propagation equation is

$$\vec{x}_t = \frac{\nabla\phi}{|\nabla\phi|} \tag{5.2}$$

replacing $\vec{x}_t$ by a finite difference scheme, Eqn. (5.2) can be rewritten as

$$\vec{x}^{n+1} = \vec{x}^n \pm \frac{\nabla\phi}{|\nabla\phi|}dt \tag{5.3}$$

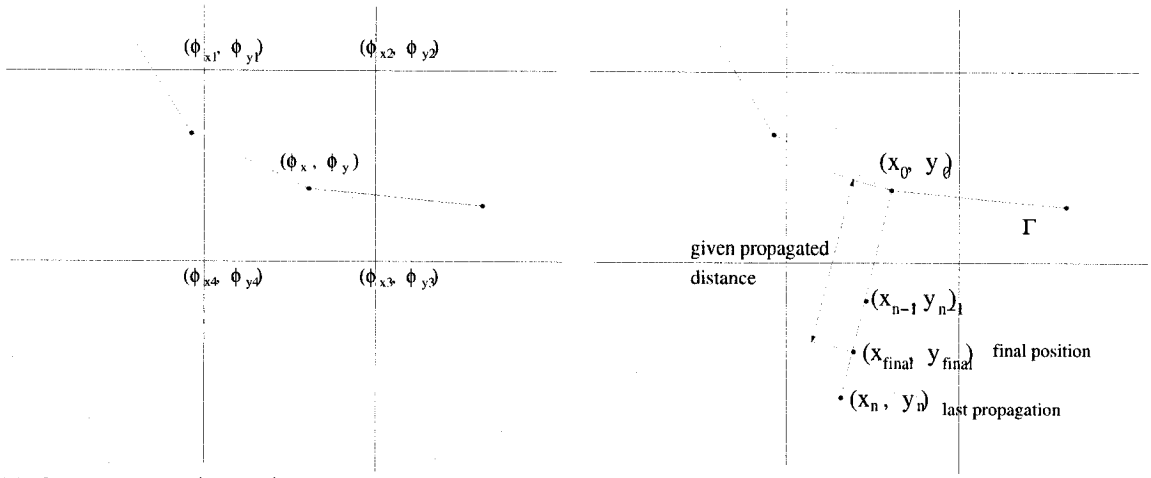
The term $\frac{\nabla\phi}{|\nabla\phi|}$ can be viewed as a unit propagation speed in the normal direction, with a positive value for an outward propagation, and a negative value for an inward propagation. The propagation process can be divided into three steps:

1. calculate ϕ_x, ϕ_y, and ϕ_z at each Cartesian grid node using a centered finite difference scheme for internal nodes, and one-sided finite difference scheme for the nodes located at the boundary of Ω. Then, calculate ϕ_x, ϕ_y, and ϕ_z for all the points to be propagated on Γ using interpolation (Fig. 5.2(a)).

2. propagate points on Γ using a fourth-order Runge-Kutta method. Propagation is stopped when the value of ϕ at position $\vec{x}^{n+1}$ is greater or equal to the given propagation distance. If the value of ϕ at the final position $\vec{x}^{n+1}$ is greater than the given propagation distance, linear interpolation is used to determine the final position $\vec{x}_{final}$ as shown in Fig. 5.2(b).

3. construct Γ_p by sequentially connecting all points on Γ_p using the parameterization information conveyed from Γ.

The 4^{th} order Runge-Kutta method (RK4)



(a) Calculation of (ϕ_x, ϕ_y) at Γ for all the front points to be propagated

(b) Interpolating the position for the final point

FIGURE 5.2 Propagation process

The 4^{th} order Runge-Kutta (RK4) is specified as follows. Let y' represents an initial value problem,

$$y' = f(t, y), \tag{5.4}$$

with $f : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$, and $(t_0, y_0) \in \mathbb{R} \times \mathbb{R}$ is the initial condition. A solution to an initial value problem is a function y that is a solution to the differential equation and satisfies

$$y(t_0) = y_0 \tag{5.5}$$

Then, the RK4 method for this problem is given by the following equation:

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5.6}$$

where n is the iteration time, h is the incremental value at each interaction, and k_1 is the slope at the beginning of the interval; k_2 is the slope at the midpoint of the interval, using slope k_1 to determine the value of y at the point $t_n + h/2$ using Euler's method; k_3 is again the slope at the midpoint, but now using the slope k_2 to determine the y-value; and k_4 is the slope at the end of the interval, with its y-value determined using k_3. They are calculated by:

$$k_1 = f(t_n, y_n) \tag{5.7}$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right) \tag{5.8}$$

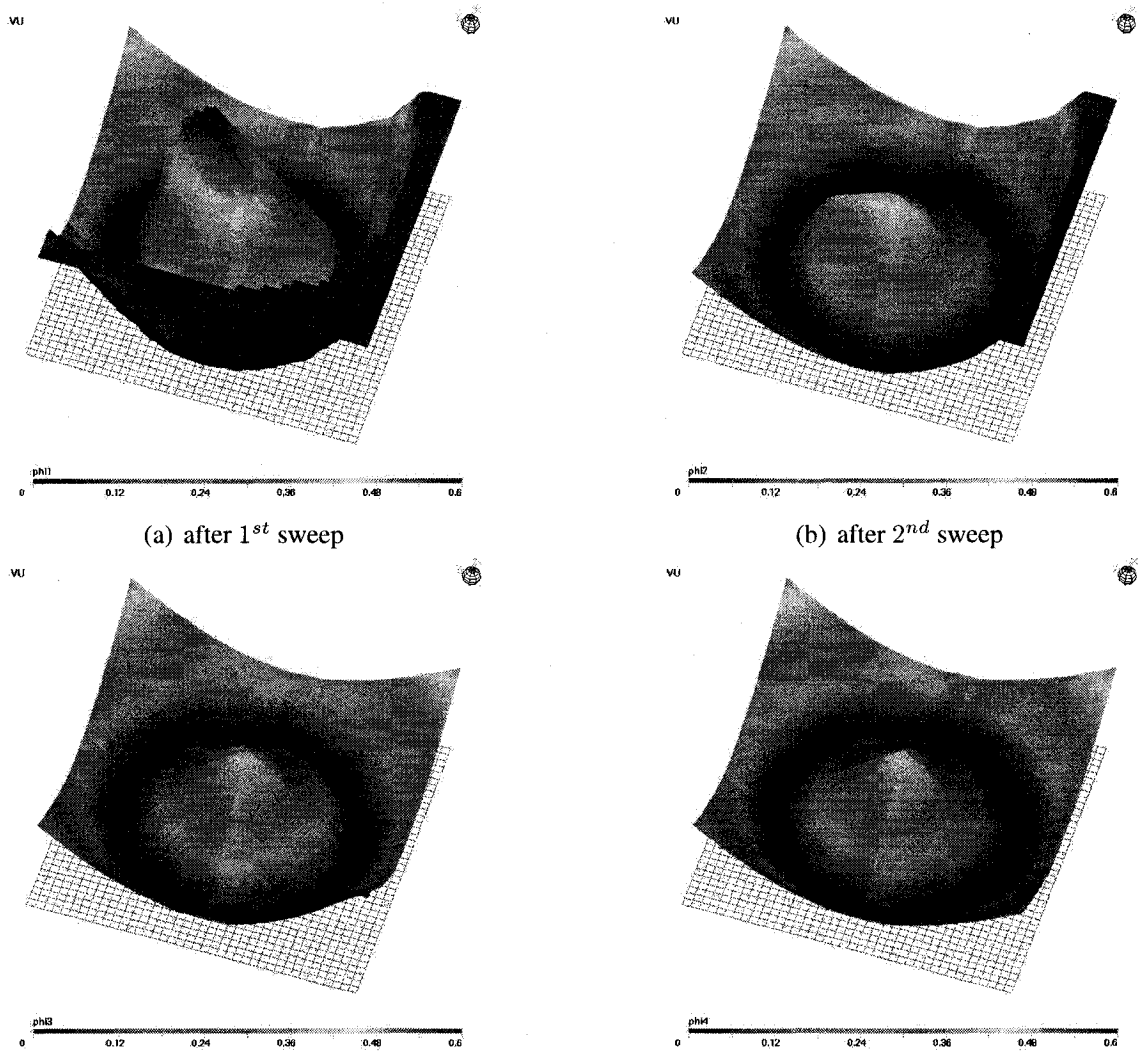
$$k_3 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right) \tag{5.9}$$

$$k_4 = f(t_n + h, y_n + hk_3) \tag{5.10}$$

5.3 Sweeping Behavior

It is shown Zhao (2005) (*Theorem* 3.6) that the iterative solution by the fast sweeping algorithm converges monotonically to the solution of the discretized system. Furthermore, after 2^n sweeps, the error of the scheme is bounded by $|hlogh|$, if Γ is an arbitrary set of discrete points (*Theorem* 4.3); or $hlog(1/h)$, if Γ is a smooth curve or surface (*Theorem* 4.4). Exponent n is the spatial dimension, and h is the mesh size. The proof of these theorems are presented in Zhao's paper and are directly applicable to the present work. In Section 5.4, Theorem 4.3 is validated using a sphere surface, and Theorem 4.4 is validated using a cube surface.

Fig. 5.3 is a 2D example to illustrate the sweeping behavior in the calculation of the distance function for a circle. In 2D, it takes 4 sweeps to get a converged solution, each sweep is carried along a diagonal direction of the Cartesian grid. In Fig. 5.3(a), the sweep is performed from the



(a) after 1^{st} sweep

(b) after 2^{nd} sweep

(c) after 3^{rd} sweep

(d) after 4^{th} sweep

FIGURE 5.3 Fast sweeping process in 2D

right-lower corner to the left-upper corner. After the first sweep, the data located in the left-upper quadrant is correctly updated; Figs. 5.3(b) to 5.3(d) illustrate the other three steps of the diagonal sweeping process. After each sweep, one-quadrant data located in the destination area of the sweeping direction is updated. The same sweeping procedure is applied in 3D. Since there are eight vertices in a 3D Cartesian grid, it takes 8 sweeps to update the data located in the 8 octants.

The property of the above described sweeping behavior can be applied in the parallelized algorithm. Since the sweeping process converges after 2^n sweepings, we can send the sweeping calculations to n processors simultaneously to speed up the sweeping computing time, thus the efficiency of the algorithm can be greatly improved comparing using only one processor.

5.4 Validation

This scheme has been applied to two test cases to verify the accuracy and efficiency of the method: a circle centered at $(0, 0)$ with a radius of 0.5, and a cube centered at $(0, 0, 0)$ with a side length of 0.5. The Cartesian frame size is 1.0x1.0 for the circle and 2.0x2.0x2.0 for the cube. The propagation is in the inward direction with an offset distance of 0.1, and a time increment of 0.01. The number of discretized points on the front are fixed, and the Cartesian grid size varies from 21^2 to 321^2 for the circle, and from 21^3 to 161^3 for the cube. For the circle, the front is discretized into 39 line segments; and for the cube, the front is discretized into 76800 triangles.

The purpose of these validations is to establish that the accuracy is first order, and that the complexity is $\Omega(N)$.

5.4.1 Accuracy Validation

Propagating Tessellated Surface

To assess the accuracy and the order of convergence of the method, the L_2 and L_∞ norms (Fortin and Garon (2000)) are used to measure the error on the position of the propagated front. Table 5.1 and 5.2 give the comparison of L_2 and L_∞ for an inwardly propagated circle and an inwardly propagated cube. Fig. 5.4(a) and Fig. 5.4(b) show that both L_2 and L_∞ converge as the mesh size is increased. As can be seen from these results, the method converges for increasing mesh size with quasi-linear first order accuracy.

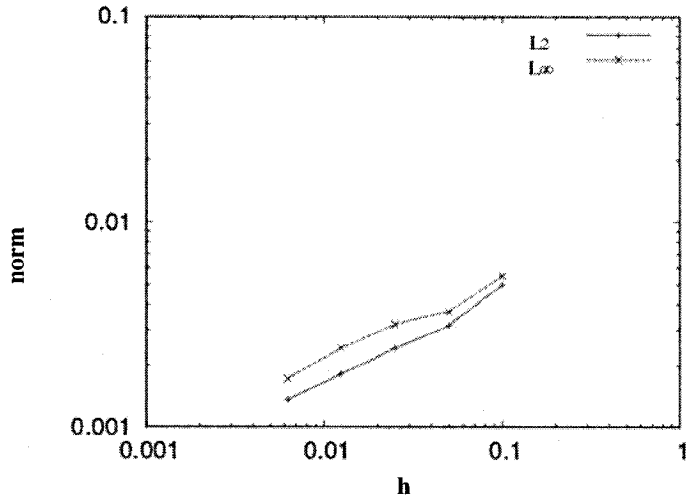
Mesh Size	21^2	41^2	81^2	161^2	321^2
L_2	0.00497535	0.00315108	0.00244957	0.00182176	0.00136155
L_∞	0.00548393	0.00367538	0.00319467	0.00246195	0.00172537

TABLE 5.1 Norm comparisons for an inwardly propagated Circle

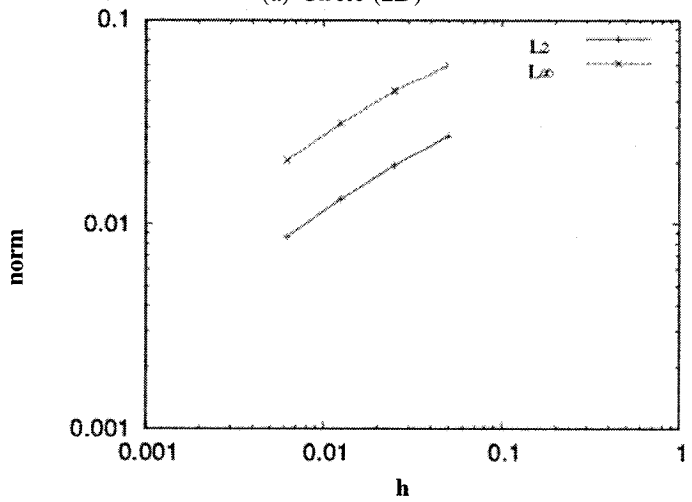
Mesh Size	21^3	41^3	81^3	161^3
L_2	0.0270416	0.01941	0.0132613	0.00864016
L_∞	0.0601622	0.0449	0.0312165	0.020511

TABLE 5.2 Norm comparisons for an inwardly propagated Cube

Fig. 5.5(a) illustrates the result of propagating a circle in its inward direction using a coarse Cartesian mesh (mesh size=21^2); and Fig. 5.5(b) illustrates the same propagation of a circle using a refined Cartesian mesh (mesh size=321^2). Fig. 5.6(b) shows the result of propagating a cube in its inward direction using a coarse Cartesian mesh (mesh size=21^3), and Fig. 5.6(c) using a refined Cartesian mesh (mesh size=161^3). From Figs. 5.5 and 5.6, we can intuitively observe that mesh size directly influences the propagated front; more precise offset surface definitions are obtained



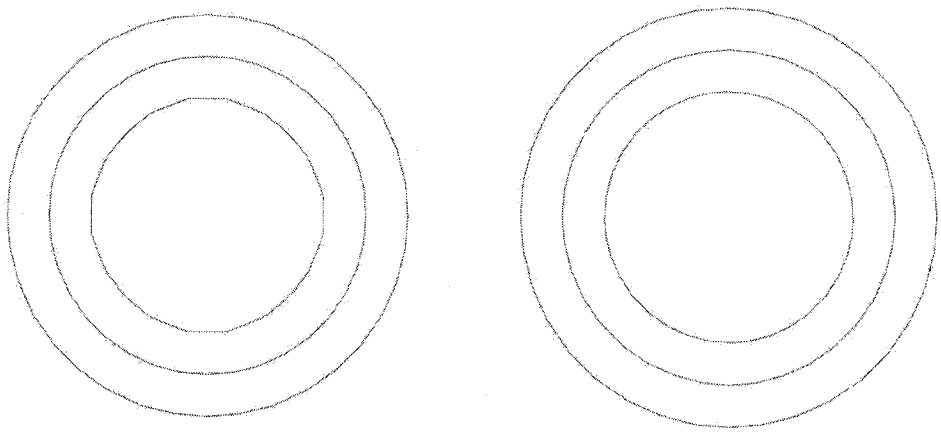
(a) Circle (2D)



(b) Cube (3D)

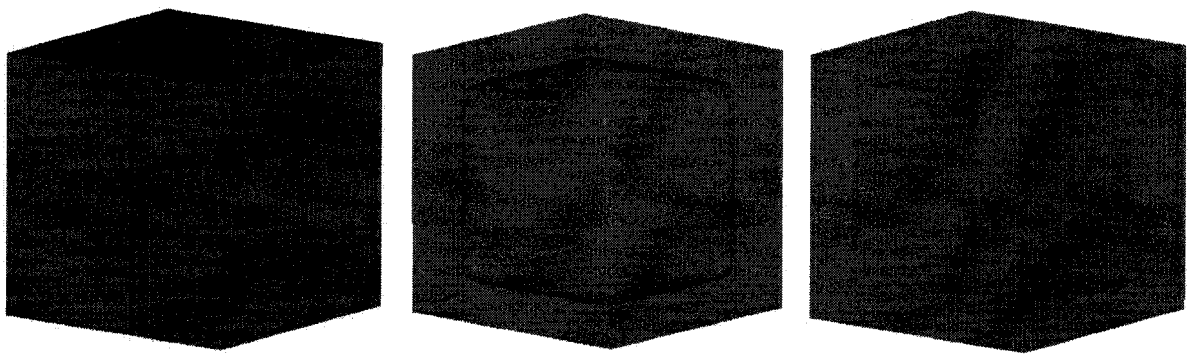
FIGURE 5.4 Norm comparisons for inwardly propagated boundaries

when Cartesian mesh density is increased. The output circle in Fig. 5.5(b) is smoother than the circle in Fig. 5.5(a), and the behavior for corners in the output cube in Fig. 5.6(c) are much sharper than the cube in Fig. 5.6(b).



(a) mesh size = 21^2 (b) mesh size = 321^2

FIGURE 5.5 Propagation of a circle (inward direction)



(a) Cube geometry (b) inward propagation (mesh size 21^3) (c) inward propagation (mesh size 161^3)

FIGURE 5.6 Propagation of a cube (inward direction)

As shown by these numerical experiments, the propagated front position directly depends on mesh size, since the unit propagation velocity in the normal direction, $\frac{\nabla\phi}{|\nabla\phi|}$, is entirely dependent on the precision of the ϕ field, which is the numerical solution to Eqn. (4.4). The fast sweeping scheme used to solve this equation is first order accurate with respect to mesh size and this dominates the

propagation accuracy.

5.4.2 Efficiency Validation

The computer used to evaluate time efficiency is under the Linux system (RedHat 9.0), CPU model name is: AMD Athlon(tm) processor; CPU: 512M, cache size: 256 KB.

In order to evaluate the efficiency of the scheme, calculation times are divided into three parts, and each is measured separately:

1. *initialization time*, which is the time used to initialize the boundary conditions, i.e. the exact distance for the points near or on the front;
2. *fast sweeping time*, which is the time used to compute the numerical solution ϕ to Eqn. (4.4);
3. *offset front obtaining time*
   - *iso-front capturing time*
     In chapter 4, we discussed that the offset front needs to be extracted from the ϕ field. In this case, *iso-front capturing time* is the time used to capture the constant-ϕ fronts, i.e. the interpolation time to compute the iso-value lines(surfaces);
   - *propagation time*
     In this chapter, we discussed that the offset front is directly obtained after propagation. In this case, *propagation time* is the time used to advance all the points located on the original front to their destination positions.

Since there are two ways to compute the *offset front obtaining time*, in this section, we will compare the results of the total consumed time using different offset front obtaining method. The total consumed time is calculated using either:

$$total\,time = (initialization)+(fast\,sweeping)+(iso-front\,capturing)\ \ (Chapter\,4) \quad (5.11)$$

or

$$total\,time = (initialization) + (fast\,sweeping) + (propagation)\ \ (this\,Chapter) \quad (5.12)$$

Tables 5.3 and 5.4 give the time in seconds consumed by each part of the overall algorithm to propagate a circle and a cube in their inward directions, respectively.

Figs. 5.7(a) and 5.7(b) give the graphical results for the time consumed by each part of the overall algorithm to propagate a circle and a cube in their inward directions, respectively . From Figs. 5.7(a) and 5.7(b), we can see that the initialization time, fast sweeping time and iso-front capturing time are dependent on the mesh size while the propagation time is dependent on the number of points for front initialization and time increment. Once these two factors are fixed, propagation time is a constant value. The reason for this is that in the propagation process, for a fixed number of initial points in the original front, the variable factor is the time increment dt. The number of time steps for the points on Γ to reach their destinations depends on dt. Thus the propagation time is not influenced by mesh size, but it is only a function of the number of points on the original front and the size of the time increment.

Mesh Size	21^2	41^2	81^2	161^2	321^2
Initialization	0.00457	0.007592	0.015335	0.040478	0.131439
Fast Sweeping	0.003768	0.023966	0.160162	1.2056	9.11685
Iso-front capturing	0.000212	0.000712	0.001863	0.005838	0.022492
Front propagation	0.016103	0.016017	0.015958	0.016986	0.017038

TABLE 5.3 Time Comparisons for an inwardly propagated circle

Mesh Size	21^3	41^3	81^3	161^3
Initialization	0.136941	1.03858	10.0974	154.029
Fast Sweeping	0.138135	2.06642	14.475	198.76
Iso-front capturing	0.009081	0.250023	5.53021	135.45
Front propagation	0.003874	0.003655	0.003835	0.003362

TABLE 5.4 Time Comparisons for an inwardly propagated cube

For 3D applications, propagation is much cheaper than iso-capturing, and this method allows to transport parametrization for both inward and outward propagations.

5.5 Applications

The purpose of this Section is only to demonstrate that the proposed propagation technique has been successfully applied in medical applications. This section illustrates a complex medical example, a carotid vessel, which was supplied by the Mechanical Engineering Department of École Polytechnique, courtesy of the *Groupe de recherche of Garon A., Thiriet M. et Farinas M.I. - ACE Project INRIA/EPM.*

The geometric model of this case has both convex and concave shape. The description of the input file is a triangulated surface, and is written in STL format. In this case, all the points are

propagated toward its inward and outward directions. The output surfaces preserve exactly the same connectivity as the input file.

5.6 Discussion

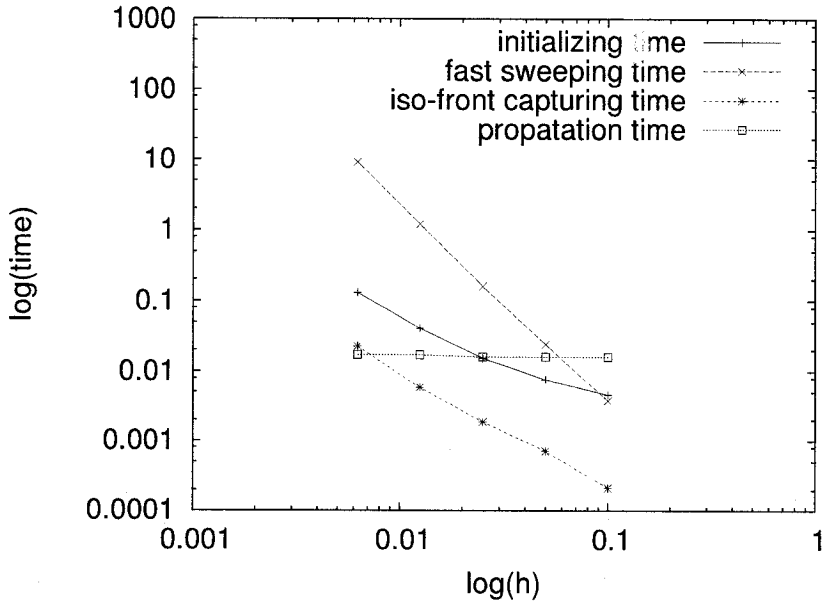
The developed offset method is successfully applied to construct offset surfaces, which can enforce the parametric consistency between the original surface and its offset. This is achieved by a combination of the traditional direct offset techniques and the present Eikonal-based approach.

The main difference between the current method and traditional direct offset methods lies in the manner in which the local normal direction is computed. In direct offset method, the local normal directions are computed using geometric information, such as the positions of neighboring points located on initial front. The central idea of the method is thus to propagate each discretized point of the original geometric front along its normal direction in ϕ space rather than geometric space. This results in a methodology that in addition to resolving the self intersection problems, will also maintain the parametric relationship between the original and its offset in most circumstances.

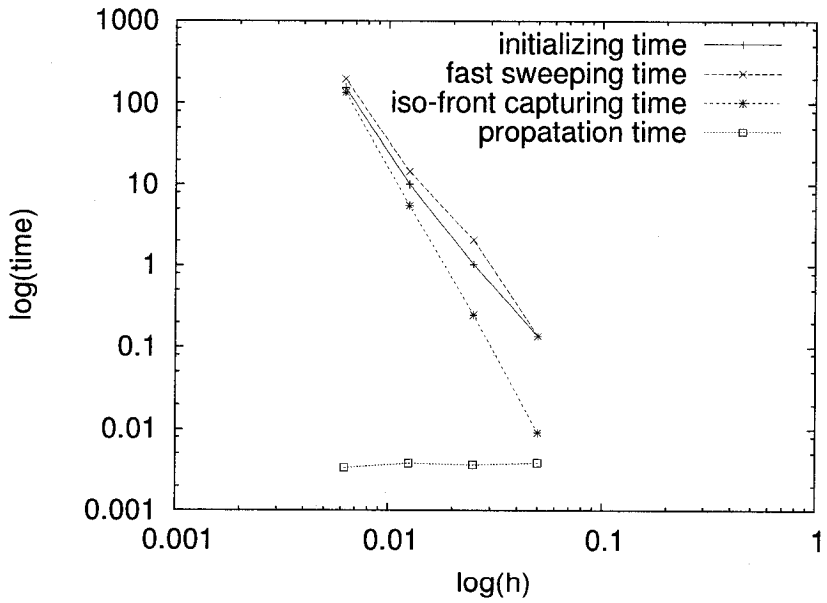
The input geometric description of the present method is tesselated surfaces (i.e., triangulated cells or quadrilateral cells), the propagation methodology is validated from accuracy and efficiency aspects respectively. The accuracy result shows that the accuracy is quasi-linearly converged to the mesh size.

One future work of this present method is to further study how to determine a proper value of dt used in the fourth-order Runge-Kutta method. If the value of dt is too small, then it will take a long time to reach the propagated distance; if the value of dt is too large, then the first propagation

may exceed the specified propagation distance.

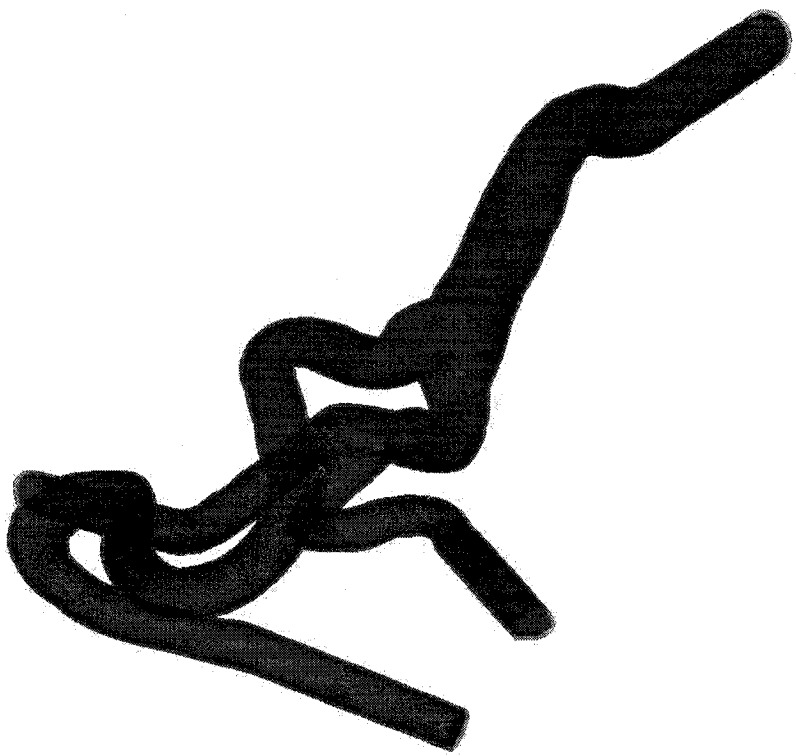


(a) Circle (2D)

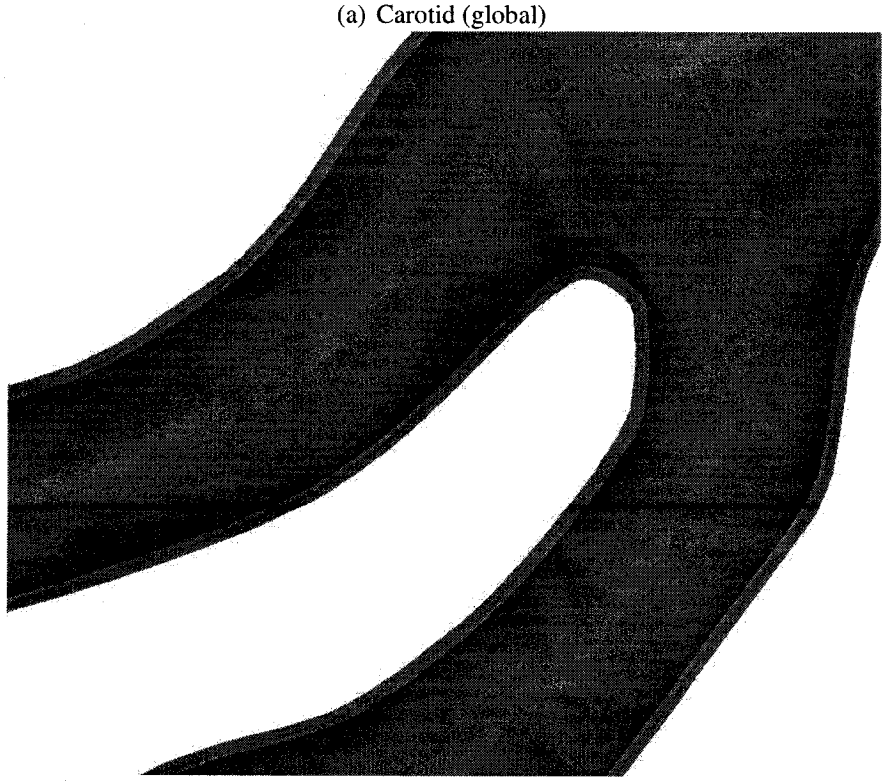


(b) Cube (3D)

FIGURE 5.7 Inward Boundary Propagation Time



(a) Carotid (global)



(b) Carotid (branch)

FIGURE 5.8 Carotid vessel

CHAPTER 6

APPLICATION 2 - BOUNDARY MESH GENERATION

This chapter discusses the use of the proposed front propagation techniques to generate high-aspect-ratio cells in the vicinity of boundaries for high Reynolds number-fluid flow simulations. Boundary dominated problems exhibit strong gradients in the direction orthogonal to the boundary compared to the other directions. This requires a minimum element size along this direction. The use of properly aligned anisotropic meshes is essential to resolve flow features, and meshes must comprise small element sizes in the direction of strong gradients and larger sizes along the others.

The present method is designed to efficiently and reliably generate good quality anisotropic semi-structured meshes near boundary surfaces for arbitrarily complex domains starting from a surface triangular mesh. First, face-matching multi-blocks emanating from an arbitrary complex boundary are generated. Then semi-structured meshes are constructed in each block and the meshes are redistributed using a Laplace-Beltrami operator at block interfaces and Laplace operator within block volumes.

6.1 Methodology

The present work is based on the front propagation technique for its strong capability of controlling the orthogonality and element size. The method can prevent self-intersections in an automatic and natural manner as presented in more detail in Chapter 4. The solution to the dilemma of choosing between hexahedra and tetrahedra is to solved by employing two families of grid elements:

stretched prismatic grid cells in boundary regions, where a wall phenomena (i.e. a boundary layer) is expected, and isotropic tetrahedral elements elsewhere. Prismatic cells are composed of triangular faces in the lateral (body-surface) directions and quadrilateral faces in the normal direction. Therefore, they can provide the geometric flexibility of unstructured as well as the orthogonality and high aspect ratio characteristics of structured grids.

Tetrahedral elements appear to be appropriate for the core of the domain, because of the irregularity shape of the regions. The triangular faces of the tetrahedra can match the corresponding triangular faces of the prisms, globally forming a conformed mesh.

This section presents the mesh generation process (as illustrated in Fig. 6.1), which includes three major steps:

1. Generation of surface patches

   The boundary (Γ) of the computational domain surface boundaries are discretized into triangulated surfaces;

2. Block interfaces and boundary mesh generation

   The distance field is computed, and the nodes located on Γ are propagated, so the multi-block interfaces and boundary mesh are generated;

3. Mesh redistribution process

   The positions of anisotropic elements are redistributed within each block.

The above procedure is applied to the 3D industrial geometric model of a draft tube previously used in Section 4.6.4.

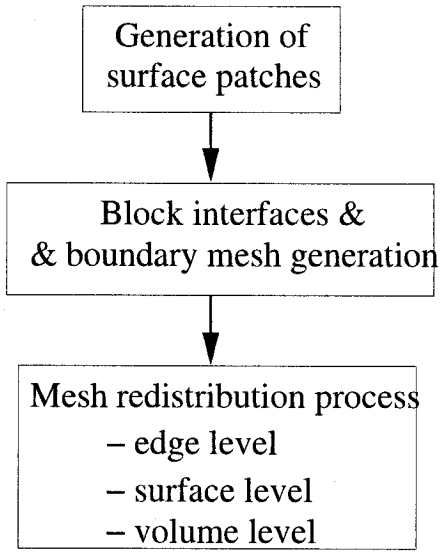


FIGURE 6.1 Boundary layer mesh generation process

6.1.1 Generation of surface patches

The first step is to decompose the domain into multiple hexahedral blocks emerging from the walls or boundary surfaces. The draft-tube surface is decomposed into a set of four-sided patches (see Fig. 6.2). These are propagated or swept into the domain, and the original patch and the propagated patch form the top and bottom faces, respectively, of a new block. A patch can originally be represented as either a triangulated surface or as a NURBS surface. In the latter case, a discretization of the patch is first constructed.

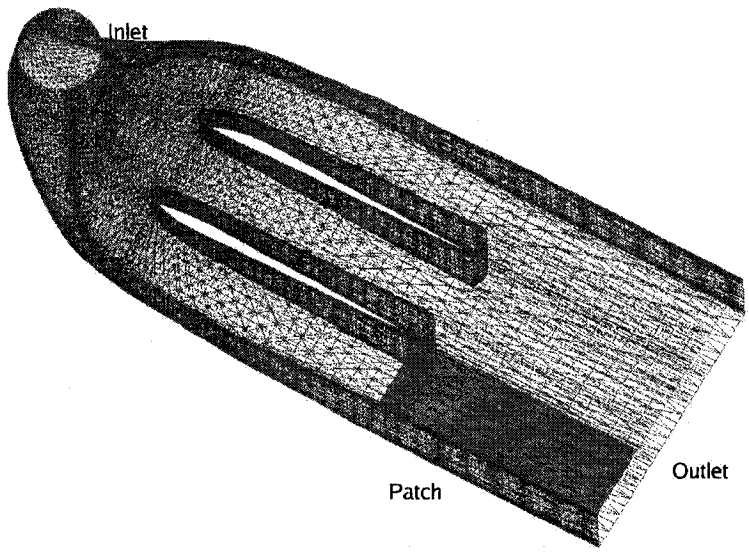


FIGURE 6.2 Geometric model: a draft tube with 2 piers

6.1.2 Block interfaces and boundary mesh generation

The offset algorithm have been defined in Chapter 5 and will not be repeated here. Two issues related to boundary mesh generation are addressed here:

Open domain

For an open domain, in addition to the boundary surfaces, it is necessary to specify the boundary curves which bound the boundary surface. For example, the inlet circle and outlet rectangle in Fig. 6.3 are such boundary curves. These lie on special boundary surfaces (planes) which are added for the purpose of correctly closing the domain from a topological point of view, as well as providing the support for the propagation of these boundary curves.

Within each such surface, the boundary curve propagation problem is transformed into a 2D

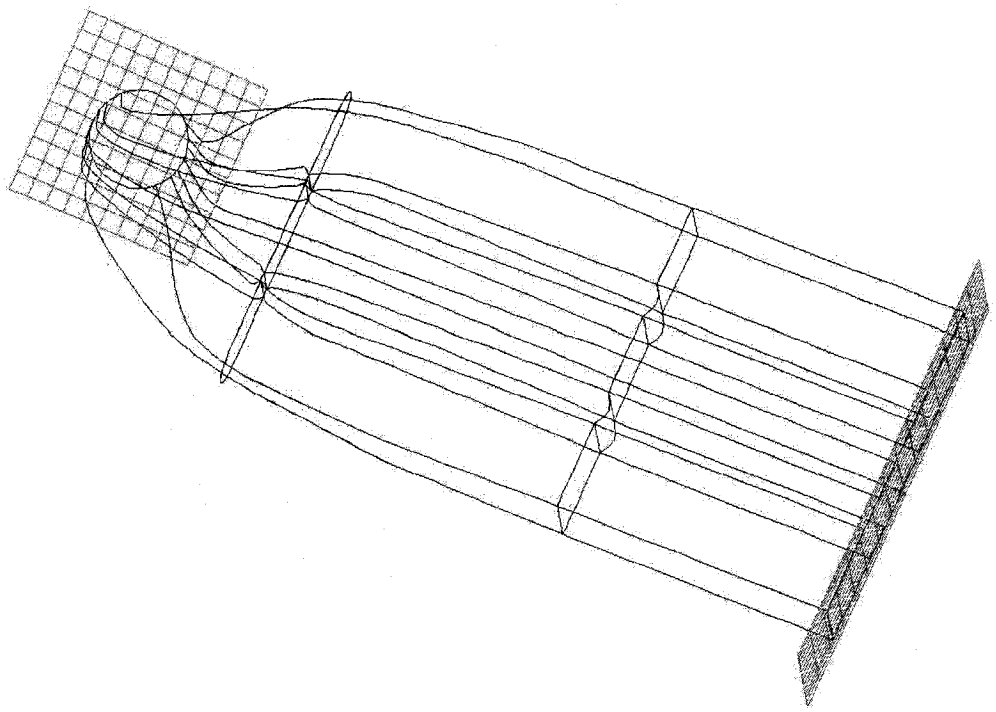


FIGURE 6.3 *Boundary curves* and *Special boundary planes*

curve propagation problem, and the algorithm described in this section is applied directly. In the present application, it is required that normal directions should lie on such special boundary planes, i.e. this algorithm cannot be used when the boundary curves are general space curves.

Block construction All the propagated points are sequentially connected according to their original connectivities, and corresponding points on the original and propagated geometries are used to construct blocks. Detailed descriptions about topological connectivities can be found in Guibault (1998).

Figure 6.4 illustrates the block construction process. Figure 6.4(a) shows the overview of

the block with the original patch surfaces in Figure 6.4(b) which form the bottom face of the block. The process of constructing the side face proceeds sequentially by connecting the propagated points of all the layers of one edge of the original patch surface as demonstrated in Fig. 6.4(c). The top face of the block in Fig. 6.4(d) is built by sequentially connecting the propagated points of the last layer according to the connectivities of the original patch surface. The final block is displayed in Figure 6.4(e).

Boundary mesh construction Boundary mesh within each block are generated simultaneously while block interfaces are constructed.

6.1.3 Mesh redistribution process

The proposed method can effectively prevent the occurrence of both global and local self-intersection problems, however, another problem arises. At sharp corners, propagation paths may be skewed together, and this can eventually cause the volume of elements connected to those paths to vanish. Fig. 6.5 illustrates the propagation behavior for the corner blocks. Fig. 6.5(a) is the overall view of two corner blocks. Figures 6.5(b) and 6.5(c) display the mesh distribution on one side face of these two blocks before applying the redistribution process.

The promising thing of the proposed method is that, local self-intersections of the propagation paths will not occur. This property guarantees that the sequence of the connectivities will always remain the same as on the original patch surfaces. This property can be used to solve the above mentioned problem, that is to redistribute the position of boundary mesh nodes to improve mesh quality. Since mesh redistribution process is a standard strategy for improving the geometric quality of a grid without modifying the topology of the mesh and therefore not requiring manipulation of the data structure. The quality of redistributed meshes is discussed in Section 7.3. The grid in

each block is redistributed by introducing grading functions. The mesh redistribution process is performed sequentially at three different levels, *edge level*, *surface level*, and *volume level*.

Edge level

At the *Edge* level, the positions of mesh points at side faces of one block are redistributed by a user defined function. In the present work, for simplicity, we redistribute the propagated points according to the ratios of edges located on the original patch surfaces. Fig. 6.6 illustrates the redistribution result. From it, we can see that, the position of propagated points are redistributed, and the propagation paths are evenly redistributed. Once the positions of mesh nodes on side faces are optimized, they are kept fixed in the later redistribution process.

Surface level

At the *surface* level, all the edges of blocks have been already redistributed at the edge level, so the four edge points on the top face are used as boundary conditions. The problem now can be viewed as, given a space surface, the nodes on the four edges are fixed, and the mesh nodes on the top face are used as the initial positions, so the Laplace-Beltrami operator (Spekreijse (1995)) is used to optimize the mesh points located on the top face of one block.

The Laplacian can be extended to functions defined on surfaces, or more generally, on Riemannian and pseudo-Riemannian manifolds. This more general operator goes by the name *Laplace-Beltrami operator*. One defines it, just as the Laplacian, as the divergence of the gradient. To be able to find a formula for this operator, one will need to first write the divergence and the gradient on a manifold. the formula for the Laplace-Beltrami operator applied to a scalar function f is, in local coordinates

$$\triangle f = div(\nabla f) \tag{6.1}$$

Volume level

At the *Volume* level, the location of the six surface meshes are fixed and used as boundary conditions, so the Laplace operator is used to iteratively redistribute the meshes inside of the block, until it converges.

6.2 RESULTING MESH

Fig. 6.7 illustrates the difference of the resulting blocks for a closed and open domain. In Fig. 6.7(a), inlet and outlet surfaces are added to Fig. 6.2. In this case, the original geometry can be viewed as a closed domain. Fig. 6.7(b) illustrates the resulting blocks generated by directly propagating the original geometry, in this case, the original domain can be viewed as an open domain, thus special boundaries are added at the inlet and outlet parts. From Fig 6.8(a) to 6.8(d), we can see that inlet and outlet surfaces are propagated and blocks are generated when the original geometry is dealt with as a closed domain; and boundary curves are propagated along the inlet and outlet planes when the original geometry is an open domain.

Fig. 6.9 is the mesh generated around a pier and at one corner of the draft tube. Fig. 6.11 is the overall view of the resulting mesh, in which a structured mesh is generated within each individual block. Fig. 6.10 is the enlarged view for the results generated around a pier and in a sharp corner, respectively.

Like the level set method, the offset front computed by the proposed method does not intersect itself, the reason for this has been explained and proved in Crandall and Lions (1984), Sethian (1999). However, marched elements can eventually collapse causing the volume of some of the

resulting elements to vanish. Elliptical smoothing of the mesh in each block is used to attenuate this problem.

The proposed boundary mesh generation method can be applied for hybrid meshing strategy. Figures 6.12 to Fig. 6.13 illustrate the hybrid meshes generated for this draft tube, with prism elements generated using the present method around boundary layers for viscous flow, and tetrahedral elements generated using a Delaunay algorithm inside of the domain.

6.3 MESH QUALITY MEASURES

The mesh generation scheme proposed here has been applied to one academic test case, cube, and a real industry geometric model, a draft tube, to verify the quality of the method. The cube is centered at $(0.5, 0.5, 0.5)$ with a side length of 1.0. The Cartesian frame size is 1.2x1.2x1.2. The propagation is in the inward direction with an offset distance of 0.01 each time, and a time increment of 0.003, five propagations have been performed. The number of discretizing points on each cube surface is 31x31 grids, there are 1800 triangles on each cube face. Two criteria are used to evaluate the quality of the resulting mesh: *volume distribution* and *Normalized Equiangular Skewness, (Q_{EAS})*.

6.3.1 Volume distribution

The volume of one mesh element, which nevertheless is always positive, remains quite acceptable as one important criterion. As mentioned in Section 6.1.3, at sharp corners, propagation paths may bend together, this can eventually cause the volume of corresponding elements to vanish. One of

the major targets of the presented method is to improve the distribution of volume for each element.

Volume Distribution (before optimization)		
volume range	numbers	% of total number
0.0 to 0.1	12,056	22.33
0.1 to 0.2	24,389	45.16
0.2 to 0.3	7,404	13.71
0.3 to 0.4	2,252	4.17
0.4 to 0.5	532	0.99
0.5 to 0.6	101	0.19
0.6 to 0.7	399	0.74
0.7 to 0.8	95	0.18
0.8 to 0.9	2,970	5.50
0.9 to 1.0	3,802	7.04
	54,000	100.00
Measured minimum value: 1.30573e-10 Measured maximum value: 1.0		
Volume Distribution (after optimization)		
volume range	numbers	% of total number
0.0 to 0.1	3,223	5.97
0.1 to 0.2	17,386	32.20
0.2 to 0.3	15,983	29.60
0.3 to 0.4	10,603	19.63
0.4 to 0.5	5,892	10.91
0.5 to 0.6	497	0.92
0.6 to 0.7	52	0.10
0.7 to 0.8	43	0.08
0.8 to 0.9	123	0.23
0.9 to 1.0	198	0.36
	54,000	100.00
Measured minimum value: 0.0315601 Measured maximum value: 0.96		

TABLE 6.1 Mesh volume distribution results (Cube)

Table 6.1 gives the volume distribution results before and after using the mesh redistribution al-

gorithm on the cube. Volume is measured by Gambit (Fluent Inc.). Fig. 6.14 shows the graphic histogram of the volume distribution results.

6.3.2 Normalized Equiangular Skewness (Q_{EAS})

The deformation is quantified by the Normalized Equiangular Skewness, which is an indicator of the commercial mesh generator TGrid developed by Fluent Inc. It determines how close to ideal (i.e., equiangular) an element is. Highly skewed elements are unacceptable because the equations being solved assume that the cells are relatively equiangular. In the 2D cases, an element can be a triangle, or a square; in the 3D cases, an element can be a pyramid, a prism, a tetrahedron or a hex. This indicator is computed using the following formula

$$Q_{EAS} = max\left(\frac{\theta_{max} - \theta_e}{180 - \theta_e}, \frac{\theta_e - \theta_{min}}{\theta_e}\right) \tag{6.2}$$

where θ_{max} is the largest angle between the edges at each corner of an element, θ_{min} is the smallest angle between the edges at each corner of an element, and θ_e is the angle for an equiangular element (e.g., 60 for a triangle, 90 for a square).

Q_{EAS} is equal to zero for ideal elements (best), to one for badly distorted elements which are completely degenerate elements. In general, quality grids have an average skewness value of approximately 0.1 for 2D and 0.4 for 3D according to TGrid.

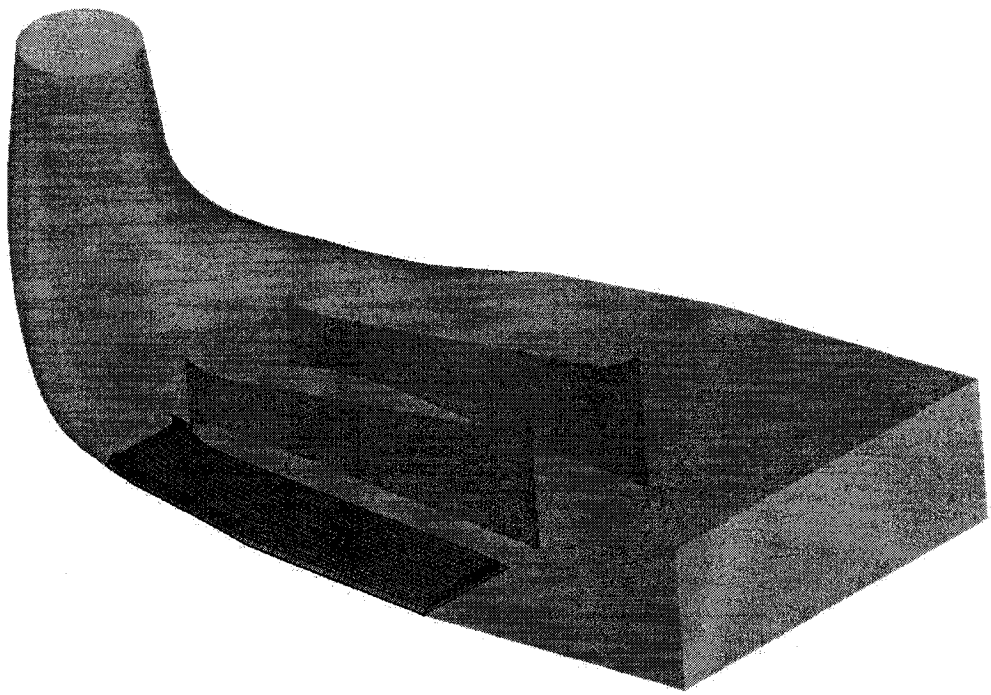
Fig. 6.15 is the histogram results of skewness for a cube. From this figure we can see that the skewness of mesh elements are uniformly distributed after mesh redistribution algorithm. The average skewness is 0.08847016 for the cube before mesh redistribution algorithm; and 0.3852097 after mesh redistribution algorithm. The result falls inside of the recommended skewness range

suggested by TGrid.

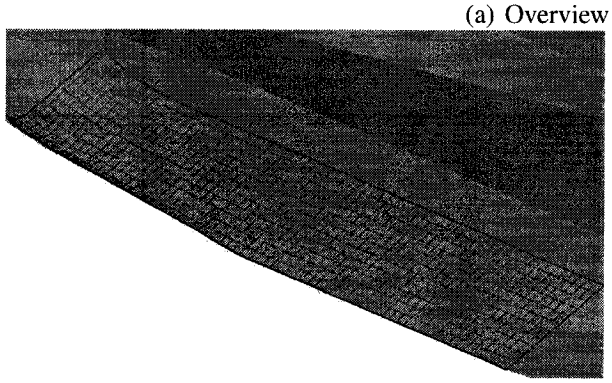
6.4 Discussion

This Chapter proposed a method to generate high-aspect-ratio cells in the vicinity of boundaries for high Reynolds number-fluid flow simulations using the new front propagation techniques. The geometric boundary is first decomposed into patch represented surfaces with triangular or quadrilateral elements, then face-matching multi-blocks emanating from an arbitrary complex boundary are generated. Finally semi-structured meshes are constructed in each block and the meshes are redistributed using a Laplace-Beltrami operator at block interfaces and Laplace operator within block volumes. The quality (i.e. volume distribution and Skewness) of the resulting mesh is measured, and it proves that the quality of the boundary mesh is acceptable.

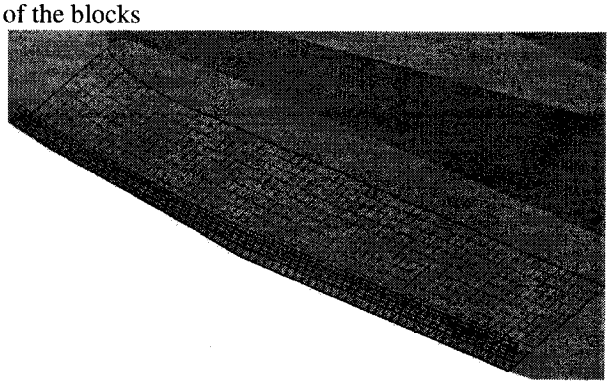
The attractive features of this method include: 1) it can be applied to very complicate geometric model, for both open and closed cases. 2) it provides a strong capability of controlling the orthogonality and element size. 3) The method can prevent self-intersections in an automatic and natural manner.



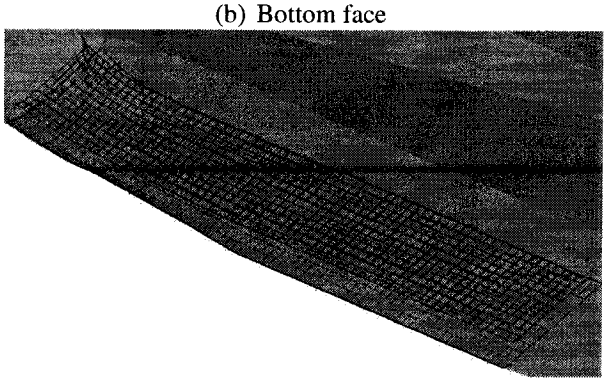
(a) Overview of the blocks



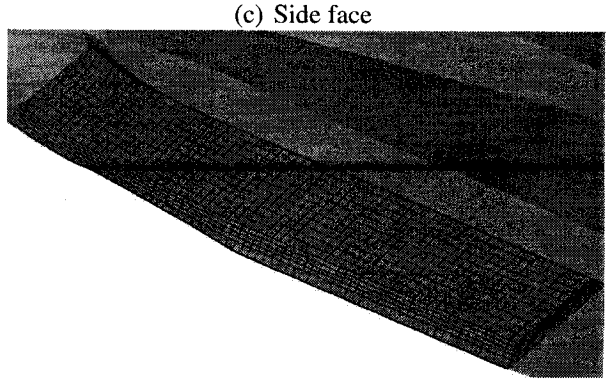
(b) Bottom face



(c) Side face

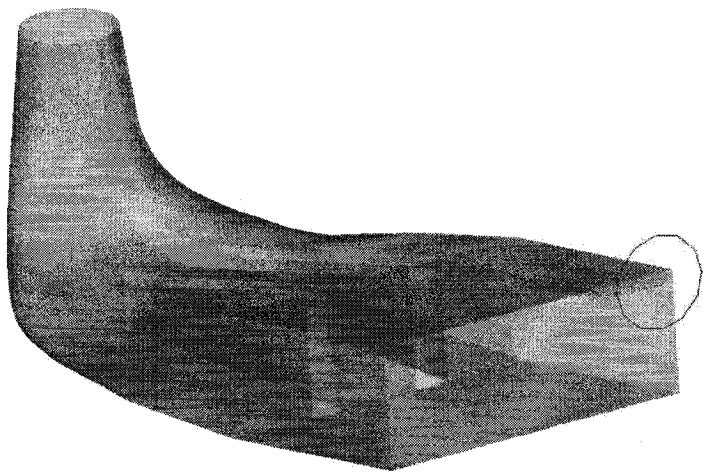


(d) Top face

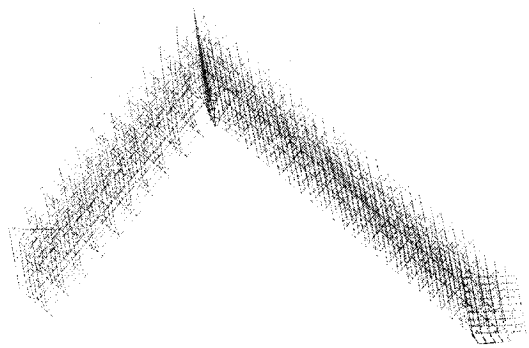


(e) Whole block

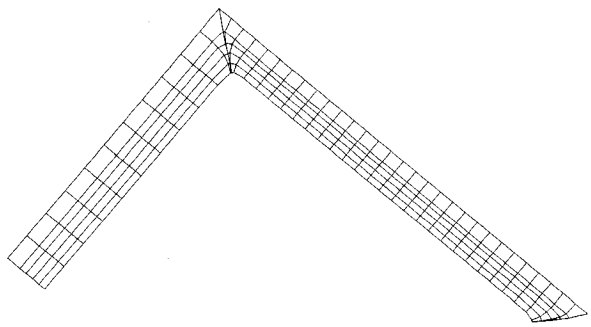
FIGURE 6.4 Corner mesh without redistribution



(a) Overview of two corner blocks



(b) with volume mesh



(c) without volume mesh

FIGURE 6.5 Before redistribution

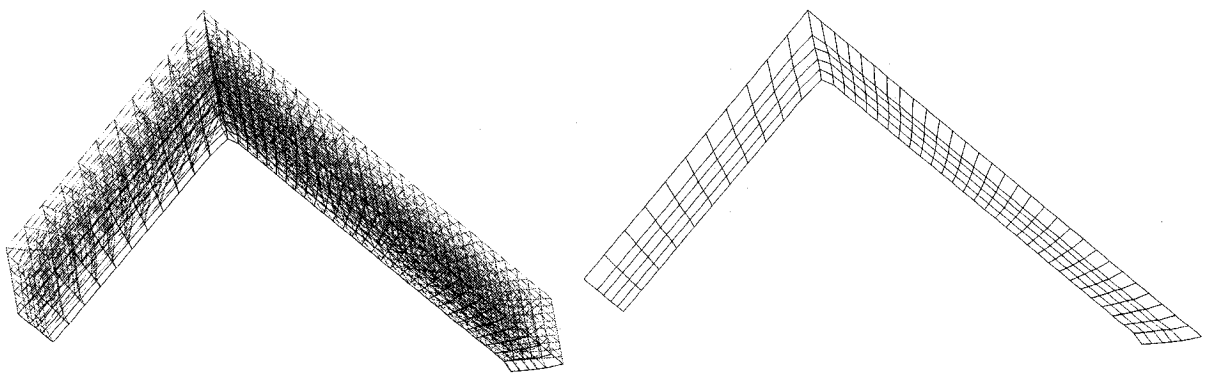
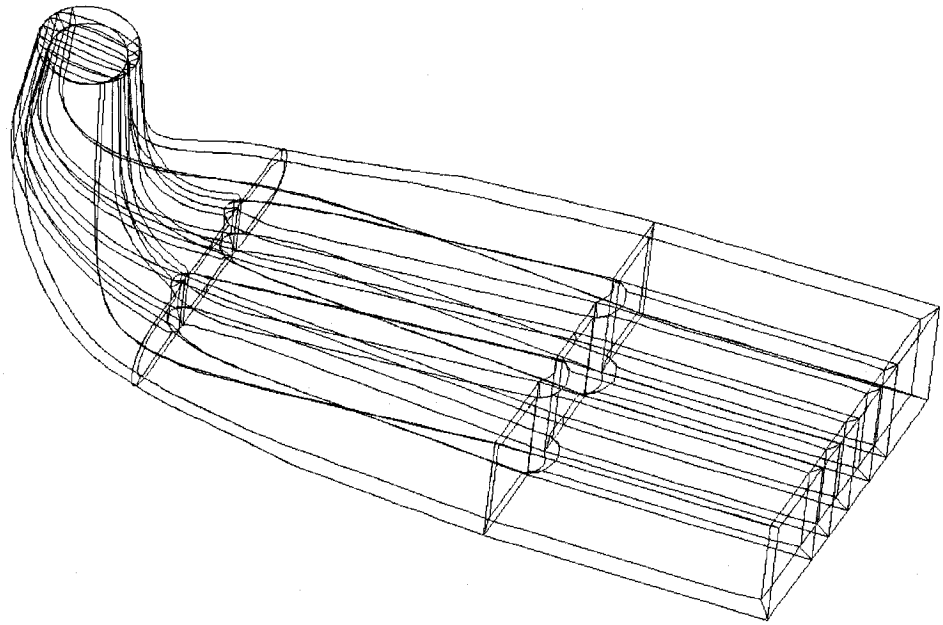
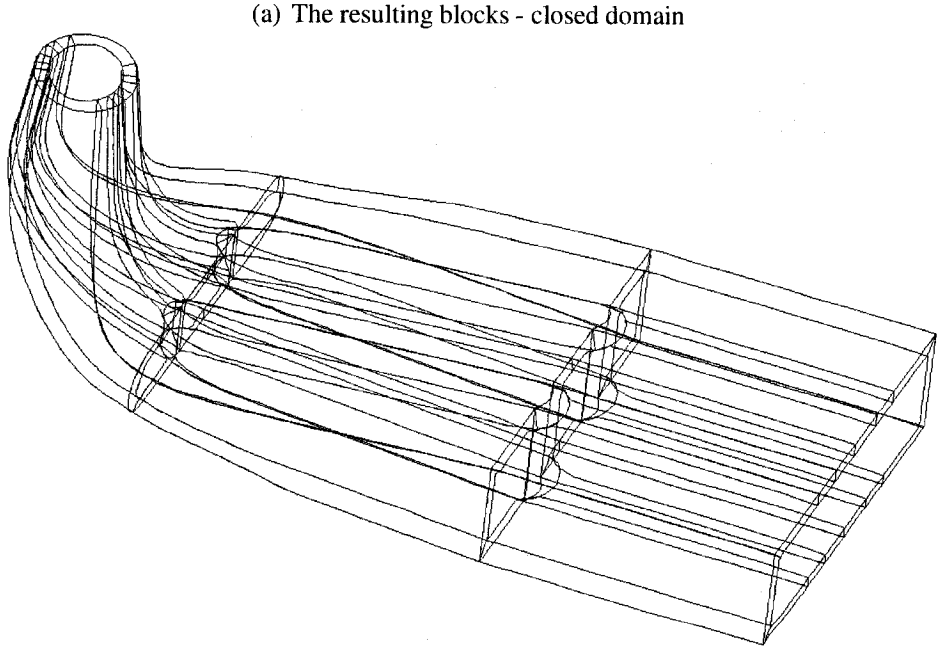


FIGURE 6.6 After redistribution

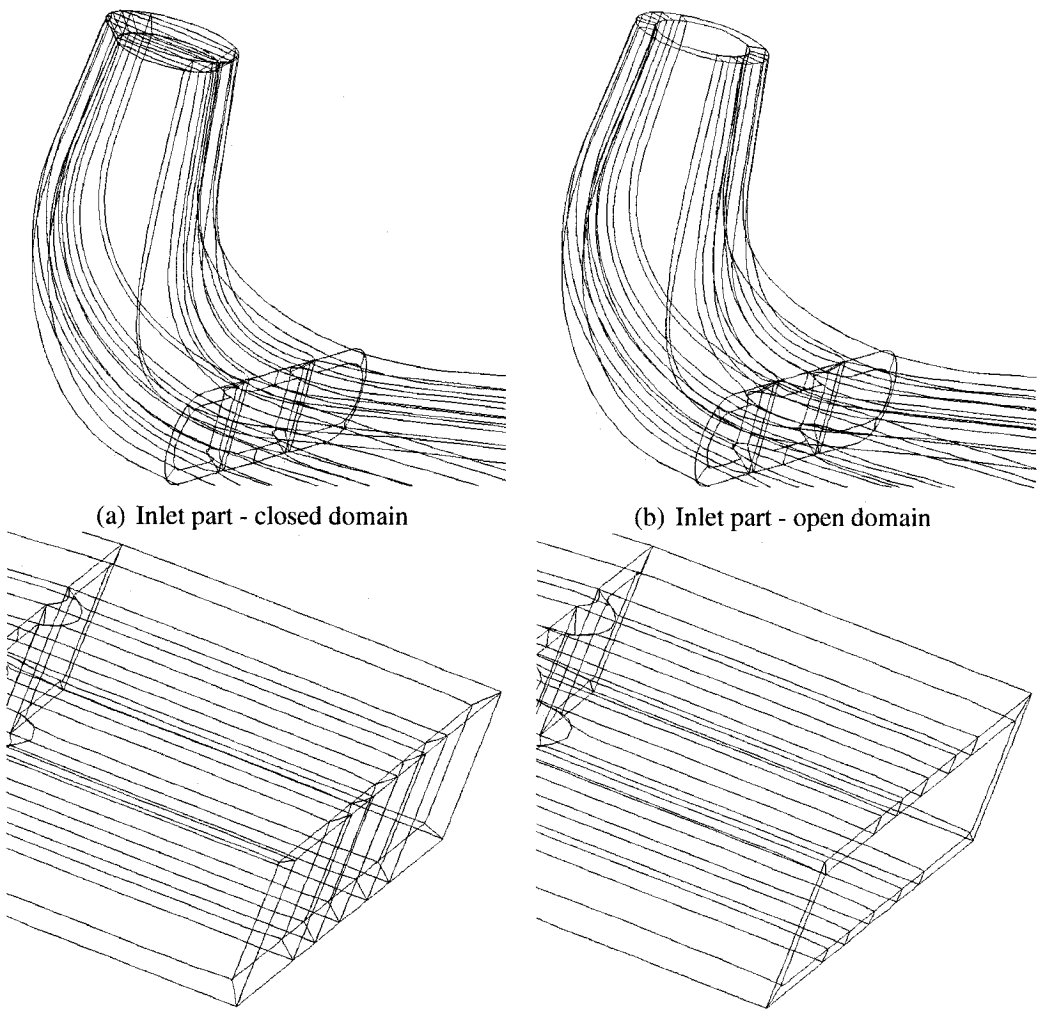


(a) The resulting blocks - closed domain



(b) The resulting blocks - open domain

FIGURE 6.7 Result comparisons (overall views)



(a) Inlet part - closed domain

(b) Inlet part - open domain

(c) Outlet part - closed domain

(d) Outlet part - open domain

FIGURE 6.8 Comparisons between open and closed domains (inlet and outlet part)

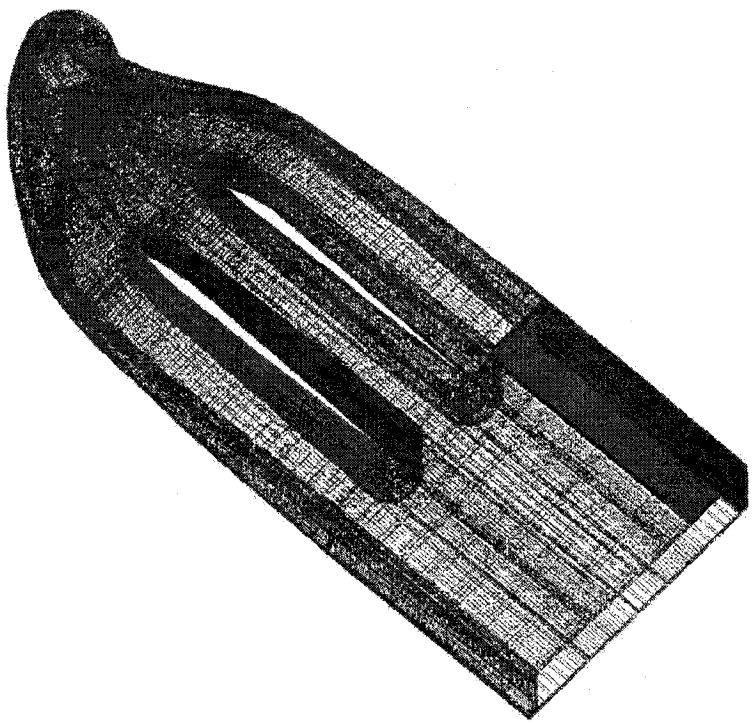
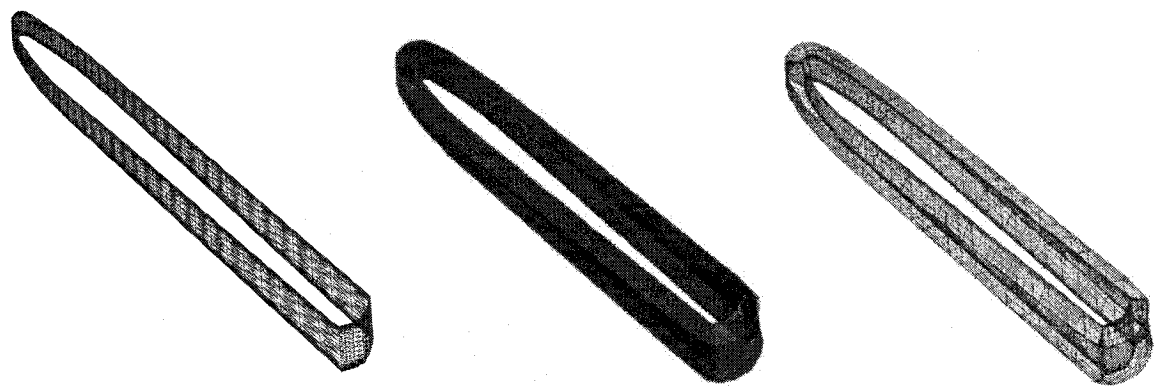
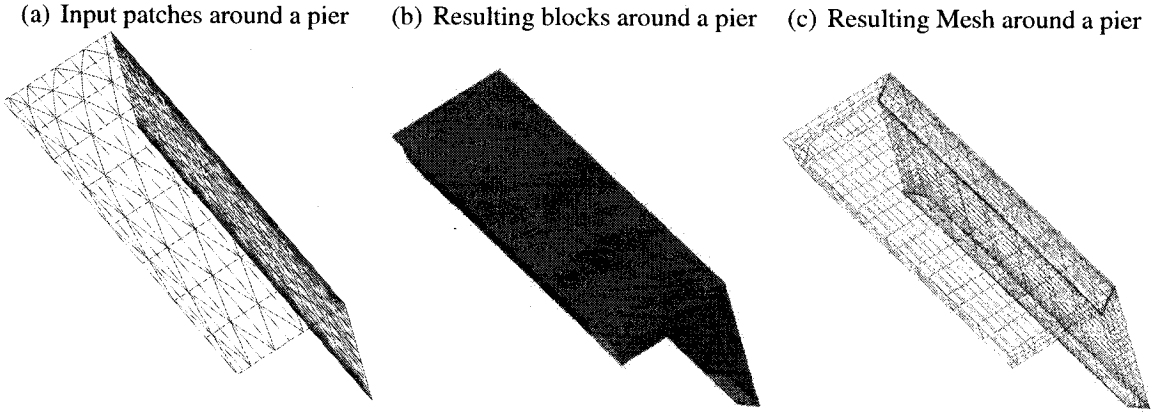


FIGURE 6.9 Resulting mesh - partial view



(a) Input patches around a pier (b) Resulting blocks around a pier (c) Resulting Mesh around a pier



(d) Input patches at a sharp corner (e) Resulting blocks at a sharp corner (f) Resulting mesh at a sharp corner

FIGURE 6.10 Resulting mesh - partial views

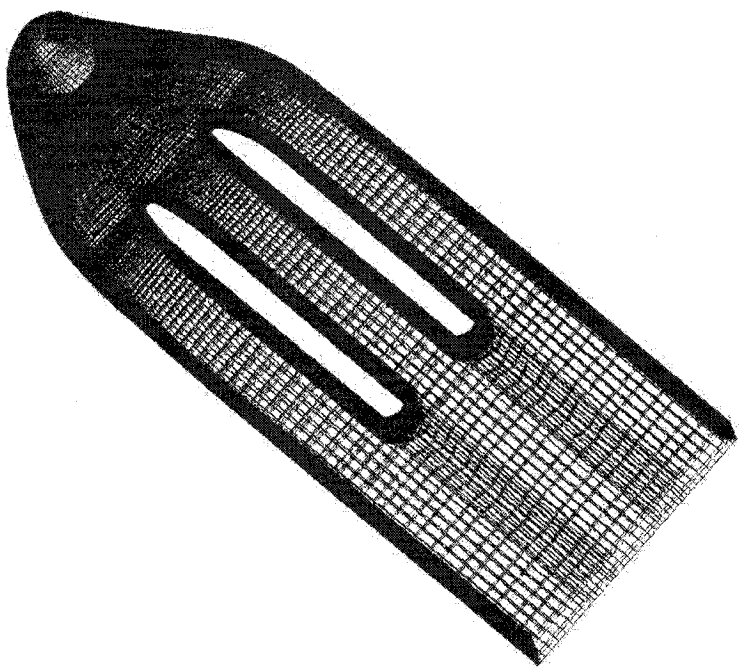
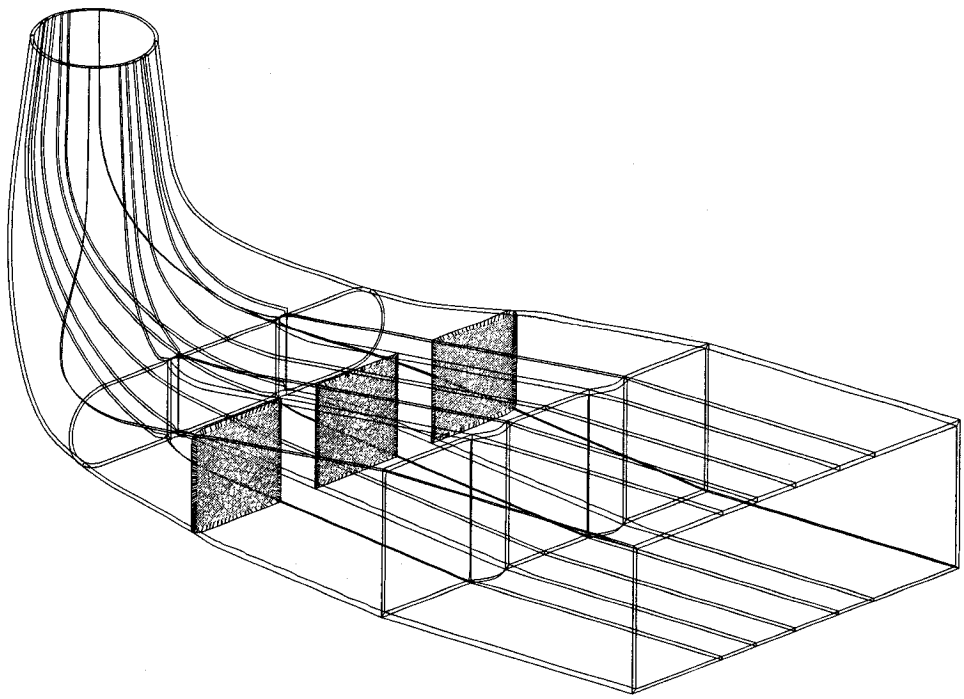
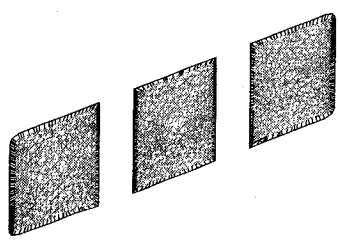


FIGURE 6.11 Resulting mesh - overall view

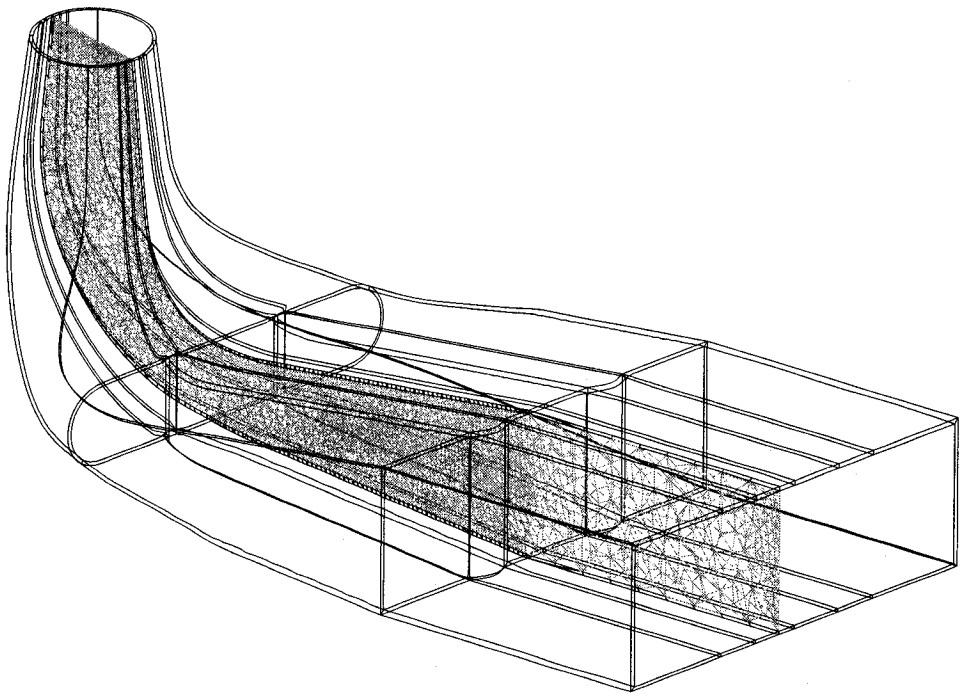


(a) Side view 1

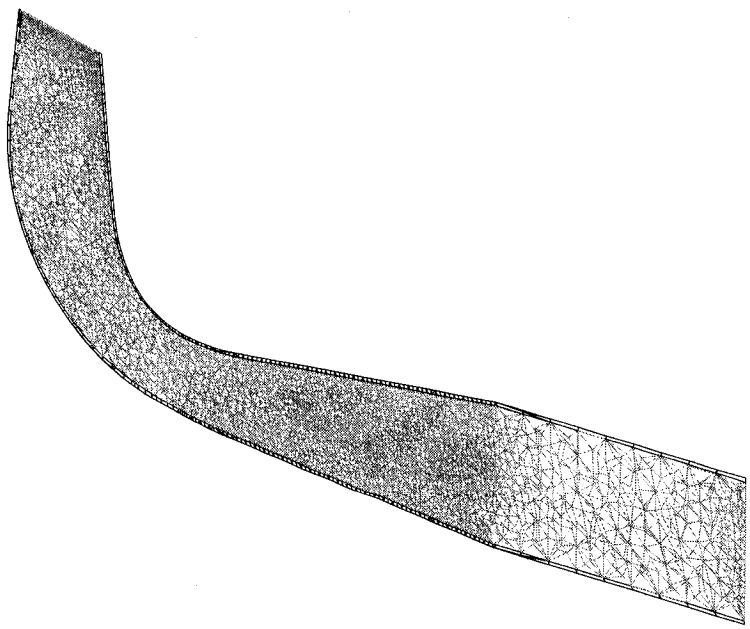


(b) Side view 2

FIGURE 6.12 Hybrid mesh for draft tube



(a) Side view 3



(b) Side view 3

FIGURE 6.13 Hybrid mesh for draft tube

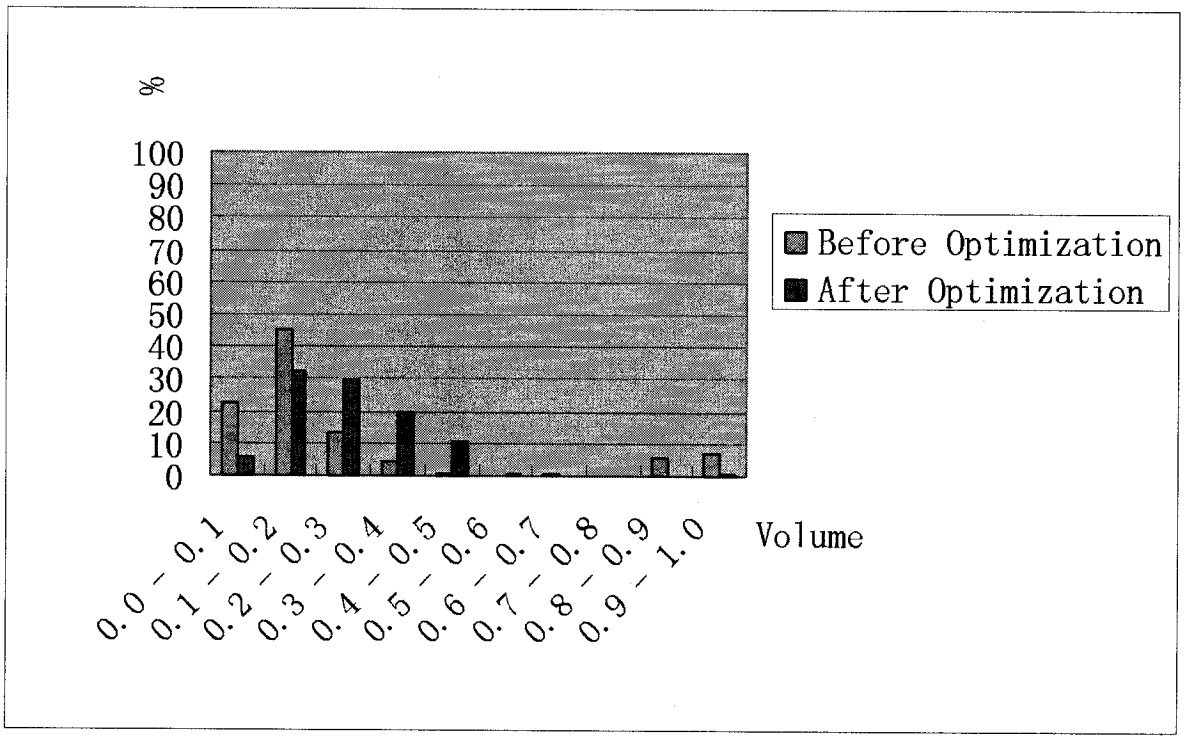
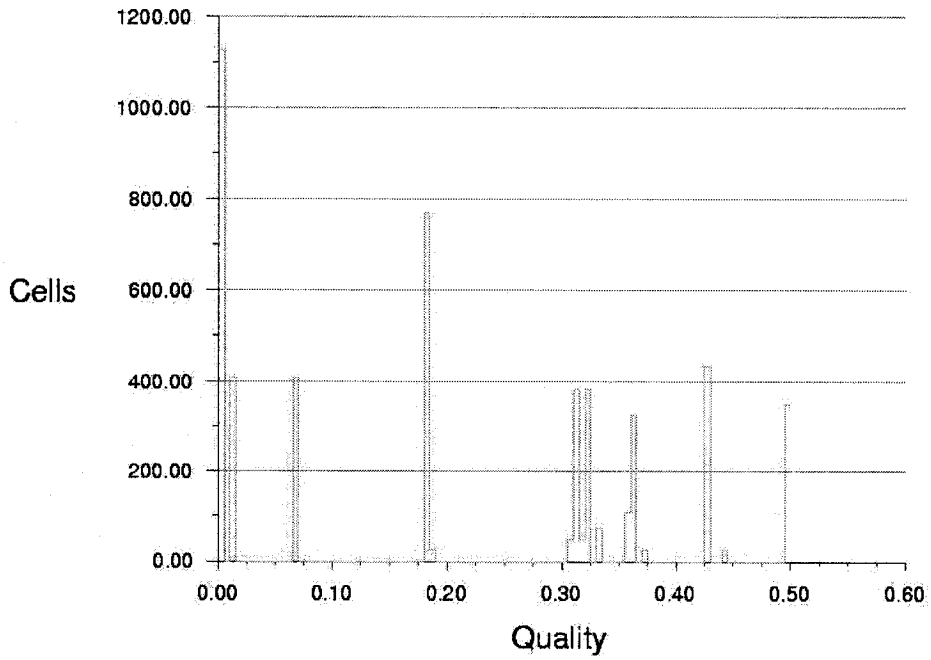
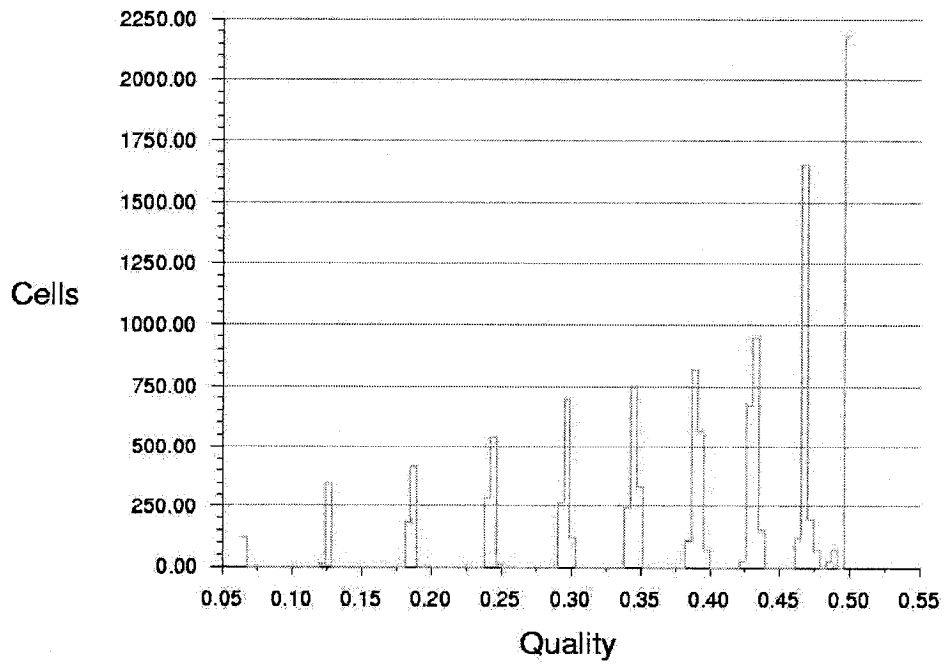


FIGURE 6.14 Histogram of the volume distribution results (Cube)



(a) Before redistribution



(b) After redistribution

FIGURE 6.15 Histogram of Skewness (Cube)

CHAPTER 7

FLUID FLOW CALCULATIONS

A great challenge for viscous flow simulations is to generate anisotropic elements in the vicinity of boundaries, especially for complex geometries. When such domains are discretized, it is important that the grid fit the boundaries, and that no conflict occur during the boundary mesh generation process.

To effectively avoid the warping of normal directions during the meshing process is the critical step to the successful generation of good quality, high-aspect-ratio cells in the vicinity of boundaries for wall dominated phenomena. For this purpose, a new approach based on hybrid meshing methodology was proposed, in which the prism elements are generated using the method described in Chapter 5.2, and the surface mesh (triangle elements on the surface) and the tetrahedral elements are produced using the ALGOR commercial softwares.

The purpose of this chapter is to validate the quality and effectiveness of this new hybrid mesh generation strategy applied to the flow in a ball valve model under both steady and unsteady conditions. The mesh quality is measured from three aspects: 1) volume distribution, 2) aspect ratio and 3) Normalized equiangular skewness. These results show that the quality of the mesh is acceptable, and the convergent fluid calculation results are illustrated in Section 7.4.

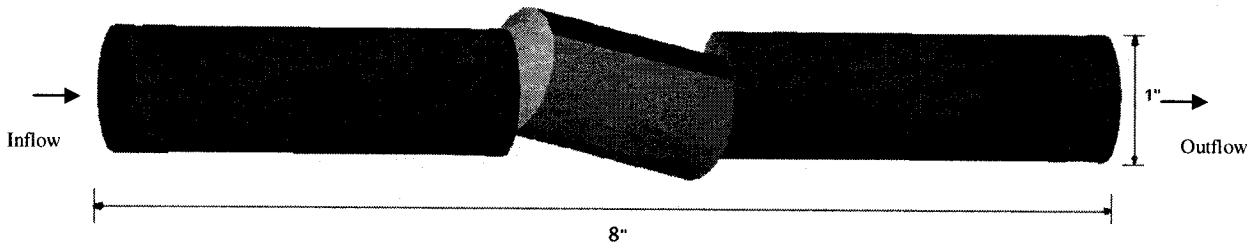


FIGURE 7.1 Diagram of the Pipe Section

7.1 Problem Description

A fluid flow simulation in a three-dimensional (3D) model of a ball valve is used to validate the proposed grid methodology. Fig. 7.1 shows the geometry of the pipe section including the ball valve which is in the half open position. The goal of the fluid flow analysis is to determine the velocity and the pressure of the fluid as it exits the section.

Topologically, the illustrated geometry is very simple and is equivalent to a cylinder. Geometrically, this configuration has both convex and concave shapes in its body. The most difficult task in this test case is to correctly calculate normal directions at the four singularity points as shown in Figure 7.2 where four surfaces pass through and share the singular point. The traditional normal calculation method uses the geometric information surrounding the point to be propagated, i.e., the weighted average normal vectors of neighboring surfaces. In such situations, the definition of the normal vector may become ambiguous when normal direction is performed in this way. For example, in Figure 7.2, the definition of normal vectors for Surface 2 and Surface 4 are almost in opposite directions, and it is difficult to determine a compromise normal vector at this point using the average normal vectors of neighboring surfaces. Instead, in the present analysis uses the proposed method Wang *et al.* (June 2005) to calculate normal vectors.

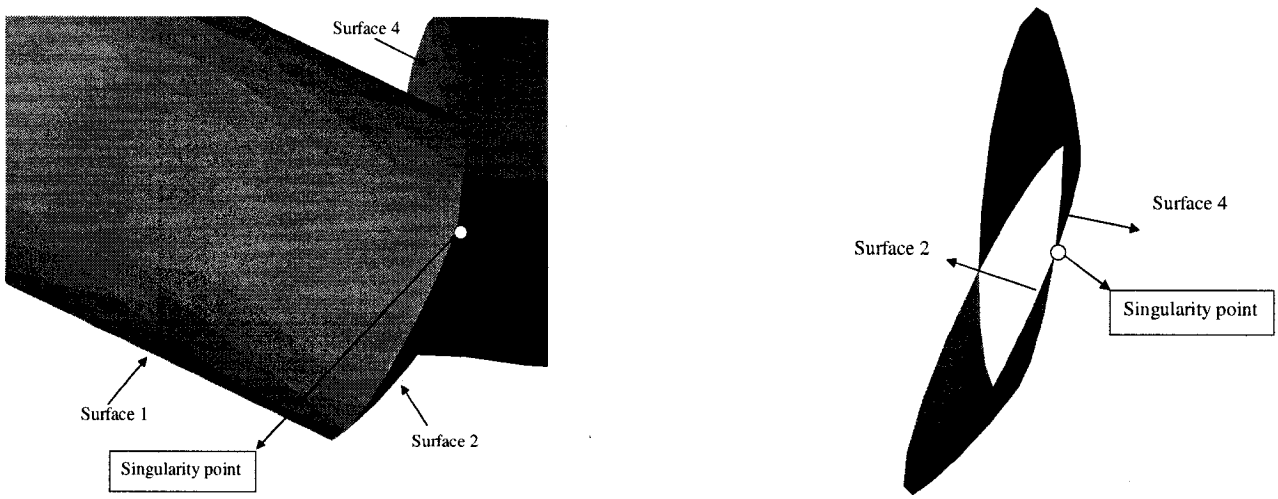


FIGURE 7.2 Singularity Point

The geometric parameters are as given in Figure 7.1. The total length of the pipes is $8.0\ in$, the diameter at intake areas and valves is $1.0\ in$. Several physical characteristics of the fluid are also pre-defined, including the flow rate at the inlet area, which is $0.5\ in/s$, the material is set as water.

7.2 Methodology

Figure 7.3 shows the outline of the proposed mesh generation process. The details for each step is explained below.

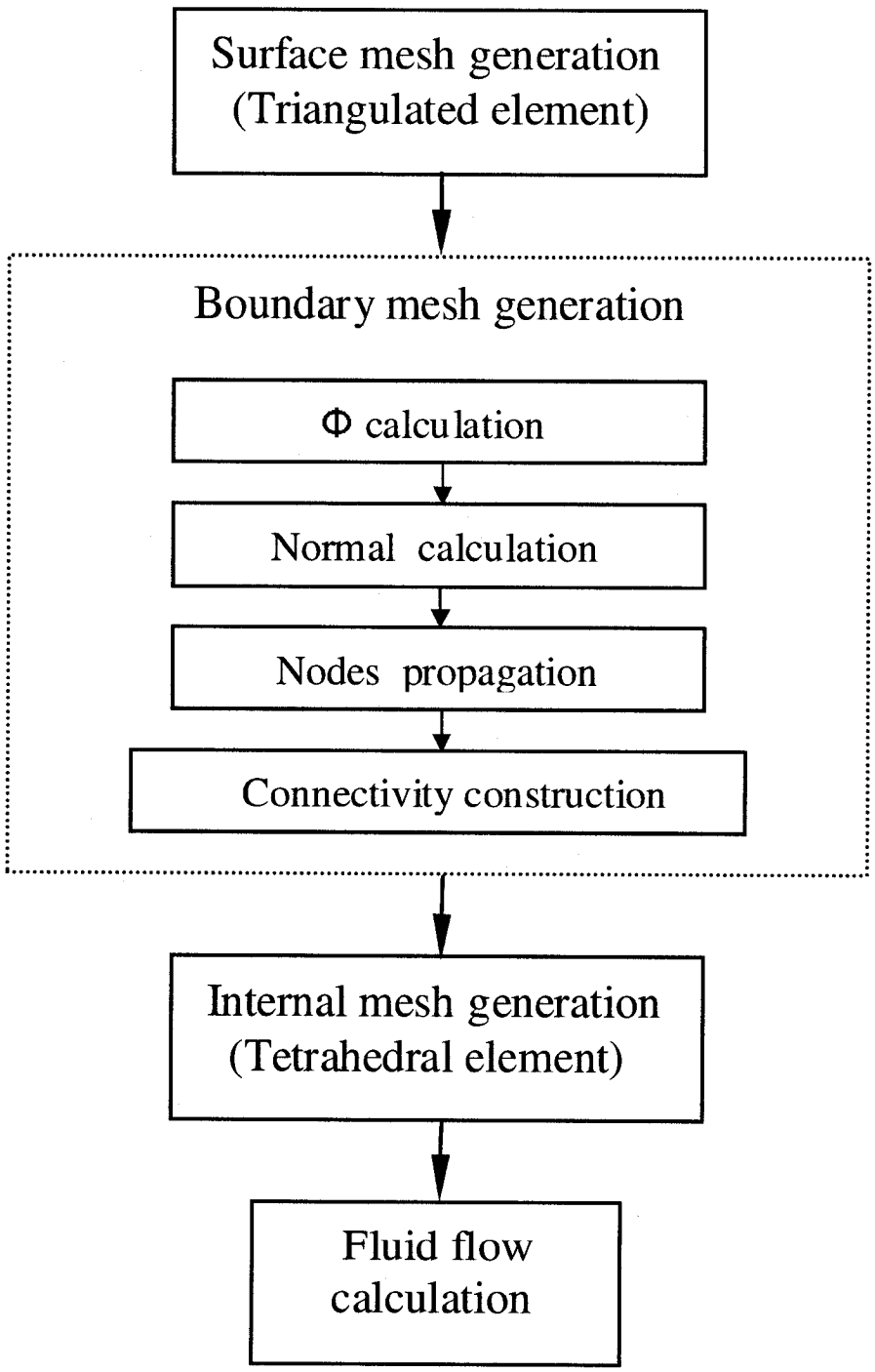


FIGURE 7.3 Outline of grid generation process

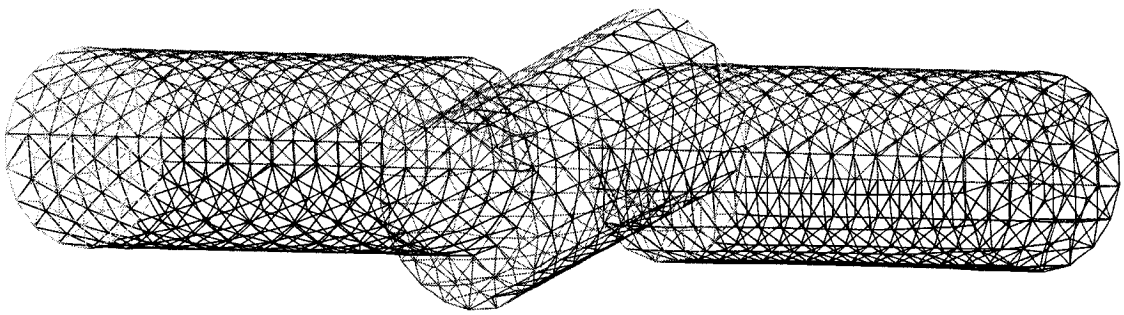


FIGURE 7.4 Surface mesh

7.2.1 Surface mesh generation

There are no restrictions for the shape of surface mesh elements. The proposed boundary mesh process can accept any type of elements, i.e., triangular, quad, etc.

In the present hybrid meshing strategy, the surfaces are discretized into triangles because tetrahedral elements are employed in the internal domain as the quality of this type of elements is easier to control. Figure 7.4 shows the initial surface mesh used in this test.

7.2.2 Boundary mesh generation

There are four steps in the boundary meshing process:

1. calculate ϕ values for each grid node using the fast sweeping algorithm. This step is explained in Chapter 4.
2. calculate normal directions for the front grid nodes, and as described in Chapter 5.
3. propagate front points along their local normal direction according to the given distance,

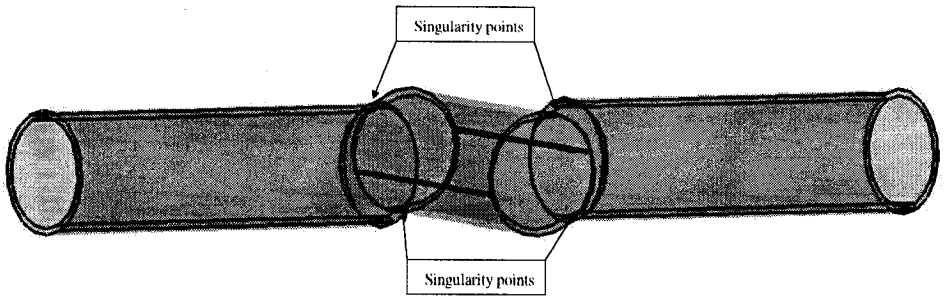


FIGURE 7.5 Boundary mesh interface

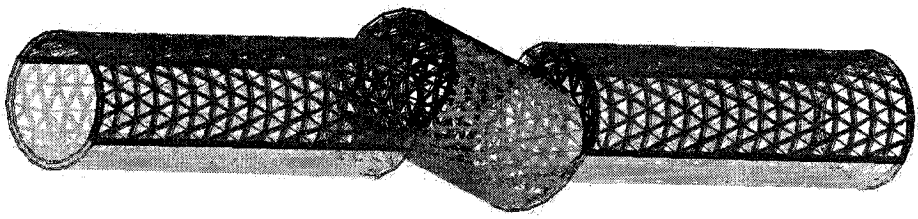


FIGURE 7.6 Boundary mesh

which is presented in Chapter 5.

4. construct blocks or the relative topological connectivity around boundary area. This algorithm is covered in Chapter 6

Figures 7.5 and 7.6 show the overall boundary mesh interface and the final boundary mesh of this ball valve model.

7.2.3 Internal mesh generation

Figures 7.7 and 7.8 illustrate the final mesh generated for this model.

After the boundary meshing process, a new surface mesh is generated which possesses exactly the

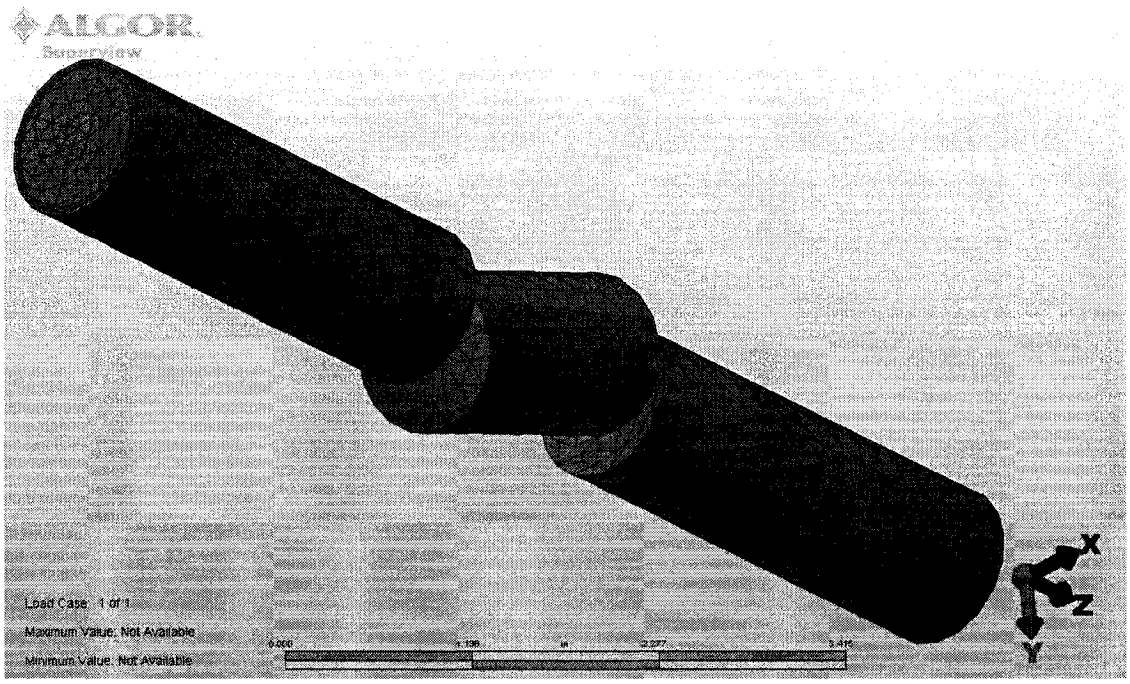


FIGURE 7.7 Final mesh (outside view)

same topological definition (triangle connectivity) as the original solid surface, and which becomes the input surface mesh for tetrahedral mesh generation. The entire computational domain defined by this surface mesh is tessellated by isotropic tetrahedra George *et al* (1991), and Borouchaki *et al* (1995).

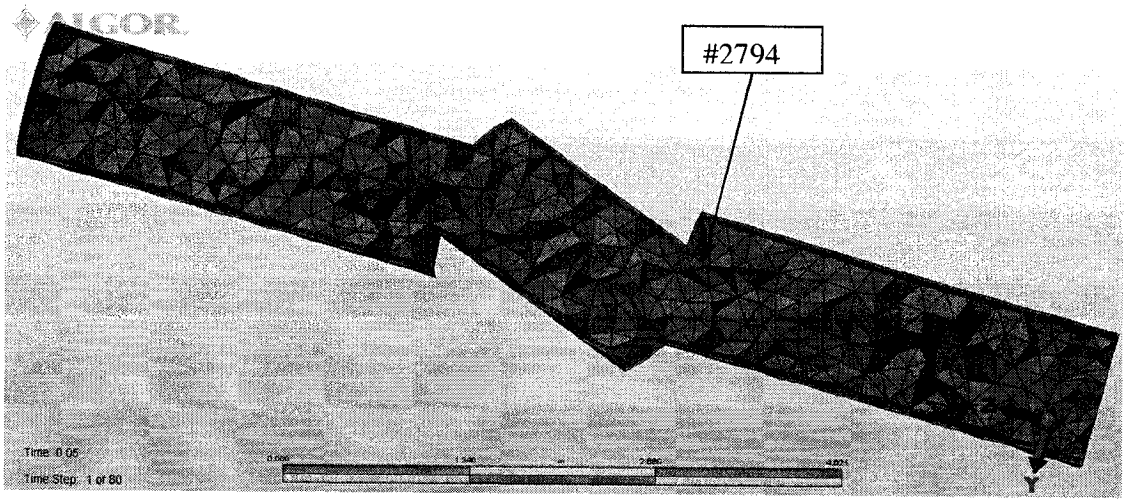


FIGURE 7.8 Final mesh (internal view)

Layer 1		Layer 2		Layer 3	
Min. Size	Max. Size	Min. Size	Max. Size	Min. Size	Max. Size
4.776e-5	4.516e-4	2.374e-5	2.851e-4	1.504e-5	2.701e-4

TABLE 7.1 Prismatic element volume range

7.3 Mesh validation

7.3.1 Volume distribution

For validation, the original boundary is decomposed into 12 surfaces, and three layers are generated after propagation. The 1-st layer is the closest layer to the valve boundary. Table 7.1 illustrates the size of the minimum and maximum volume for prismatic elements at each layer. From this Table we can see that the volume of the prisms are reduced when marching proceeds internally.

Figure 7.9 illustrates partial volume distribution of the prismatic elements. Since the normal directions on the surface mesh are defined in the outward direction of the domain, all volumes are

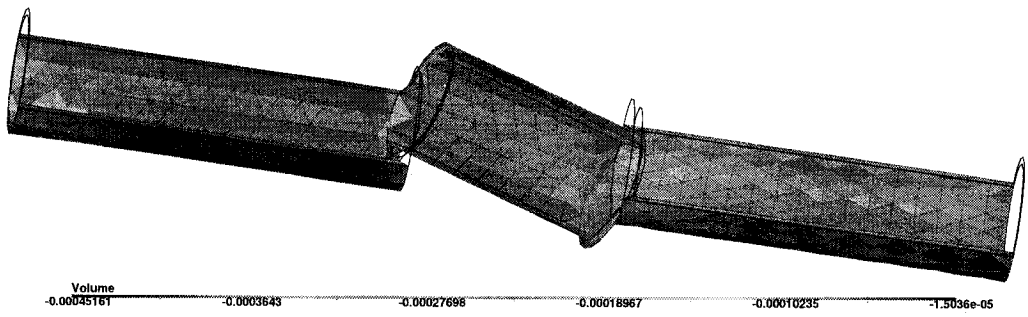


FIGURE 7.9 Volume distribution of prismatic elements

Type	Min. Size	Max. Size
Prismatic element	1.503e-5	4.516e-4
Tetrahedral element	2.904e-5	1.802e-3

TABLE 7.2 Overall volume range

labeled using negative values. In this figure, blue color represents maximum volume, and red represents minimum when absolute values are used. This figure shows that the surface mesh can greatly affect the quality of the boundary mesh. A more uniform area distribution of the original surface, will yield a better quality distribution of the prismatic elements. Table 7.2 shows the overall volume range generated for this model.

7.3.2 Aspect ratio

The definition of aspect ratio for a prism element is:

$$aspect\ ratio = \frac{averaged\ beam\ length}{averaged\ circumcircle\ radius} \tag{7.1}$$

Table 7.3 illustrates the histogram of aspect ratio for prismatic elements. From the table we can see that the averaged aspect ratio is 0.012632. The reason for such a small aspect ratio value is that the

Range	Number of prisms	%
0.003443 - 0.007660	2241	42.395006
0.007660 - 0.011877	1010	19.107075
0.011877 - 0.016093	562	10.631858
0.016093 - 0.020310	423	8.002270
0.020310 - 0.024527	362	6.848278
0.024527 - 0.028743	293	5.542944
0.028743 - 0.032960	183	3.461975
0.032960 - 0.037177	68	1.286417
0.037177 - 0.041393	64	1.210745
0.041393 - 0.045610	80	1.513432
total	**5286**	**100**
Averaged aspect ratio = 0.012623		

TABLE 7.3 Histogram of aspect ratio

Value of normalized equiangular skewness	Number of prisms	Percentage %	Cell quality
0.000000 - 0.250000	710	13.431706	Excellent
0.250000 - 0.500000	3827	72.398789	Good
0.500000 - 0.750000	391	7.396897	Fair
0.750000 - 0.900000	332	6.280742	Poor
0.900000 - 1.000000	26	0.491865	Bad
total	**5286**	**100**	
Averaged normalized equiangular skewness = 0.16386			

TABLE 7.4 Histogram of normalized equiangular skewness

initial surface mesh is very coarse. Refining the surface mesh size or increasing the offset distance will help to increase the aspect ratio.

7.3.3 Normalized equiangular skewness

The definition of normalized equiangular skewness (NES) and cell quality can be found in Fluent (2003), and it is also described in section 6.3.2. According to Fluent (2003), 0 indicates ideal elements, 1 indicates bad elements, and the acceptable NES range in 3D is 0-0.4. Table 7.4 illustrates the histogram of NES. From this table, we can see that the averaged normalized equiangular skewness is 0.16386 which falls in the acceptable range.

7.3.4 Sharp corner behavior

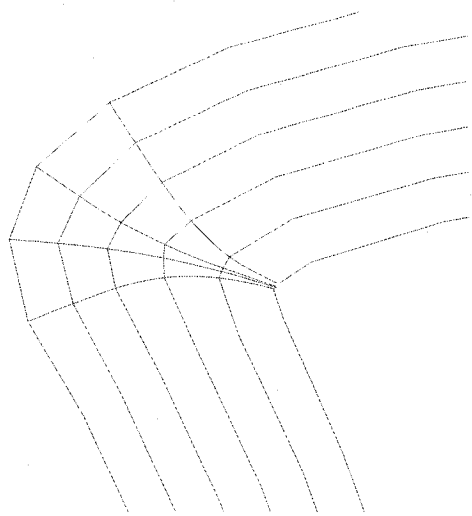


FIGURE 7.10 Propagation behavior at a sharp corner

Theoretically, this boundary mesh generation method allows the propagation to proceed to any distance without loosing the original mesh connectivity at sharp corners. However, some of the elements may degrade down to zero-volume elements when the propagated distance is greater than a certain value.

For example, Figure 7.10 shows the propagation behavior at one sharp corner in two dimensions. The boundary line is propagated in the inner direction (from left to right), and it is seen that the propagation paths are skewed together after a certain distance. A post-redistribution algorithm is required to avoid zero-volume elements during the boundary mesh generation process. Chapter 6 discusses a boundary mesh registration algorithm, which is only limited for patch decomposition of the original surface. The surface mesh of this model is a hybrid mesh (triangular and quadrilateral elements), the algorithm of how to redistribute boundary mesh generated from this kind of surface mesh is to be explored in the future work.

.

7.3.5 Singularity point propagation behavior

Figure 7.11 shows the propagation behavior at singularity points. From these figures, we can see that there is only one propagation path at the singularity point, shared by the four surface meshes. A unique normal direction is obtained at the singularity points because the ϕ value at any point in the computational domain has a unique value. Consequently the normal vector, $\frac{\nabla\phi}{|\nabla\phi|}$, is also unique at any point of the computational domain, illustrating the reason why the proposed method avoids the problem arising in the traditional normal calculation method.

7.4 Flow analysis

The purpose of performing these fluid flow analysis is NOT to compute the precise results of this flow simulation, but to achieve the following targets: 1) we want to verify the capability of

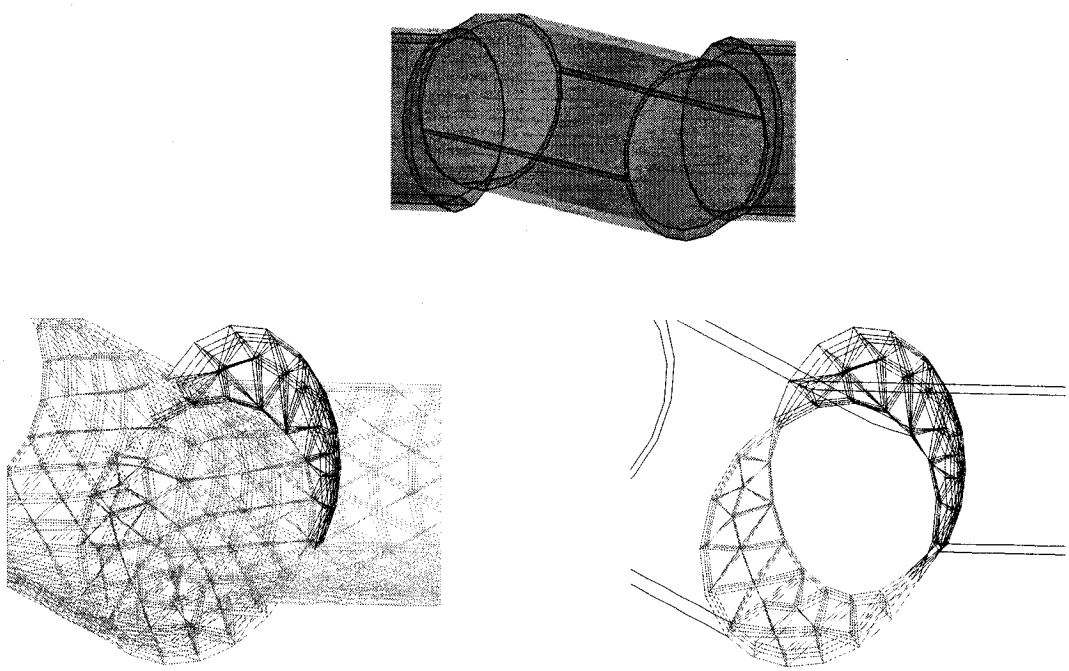


FIGURE 7.11 Mesh at singularity points

capturing vertex phenomenon at the corners of the boundary layer area; 2) if the converged velocity and pressure results can be obtained using the proposed boundary mesh. For these reasons, the original surface mesh is discretized into a very coarse mesh to illustrate successfulness of the proposed boundary meshing methodology. Also, the fluid flow analysis results are not discussed in details.

Fluid enters the pipe which is connected to a ball valve in the half open position. The Reynolds number for a pipe can be expressed as:

$$Re = \frac{\rho v d_h}{\mu} \tag{7.2}$$

where, ρ is the density (kg/m^3), v is the velocity (m/s), and d_h is the hydraulic diameter (m). Taking the water properties at temperature $20C^o$ and pressure $101325Pa$, the density is $998.29kg/ms$, and dynamic viscosity is $0.001003kg/ms$ (Thermexcel (2003)), the Reynolds number in this work is 321.

Two analysis (steady and unsteady flow) are performed using this mesh. The commercial software used in analysis is ALGOR Inc, which is a finite element analysis (FEA) solver. The velocity and pressure distributions are shown below.

7.4.1 Steady flow

The flow at the inlet is set as a uniform velocity in the direction of z-axis. The expected velocity solution in the domain is shown in Fig. 7.12. As expected, a vertex flow pattern develops in the downstream area shown in Fig. 7.13. Fig. 7.14 is the internal pressure distribution which is in

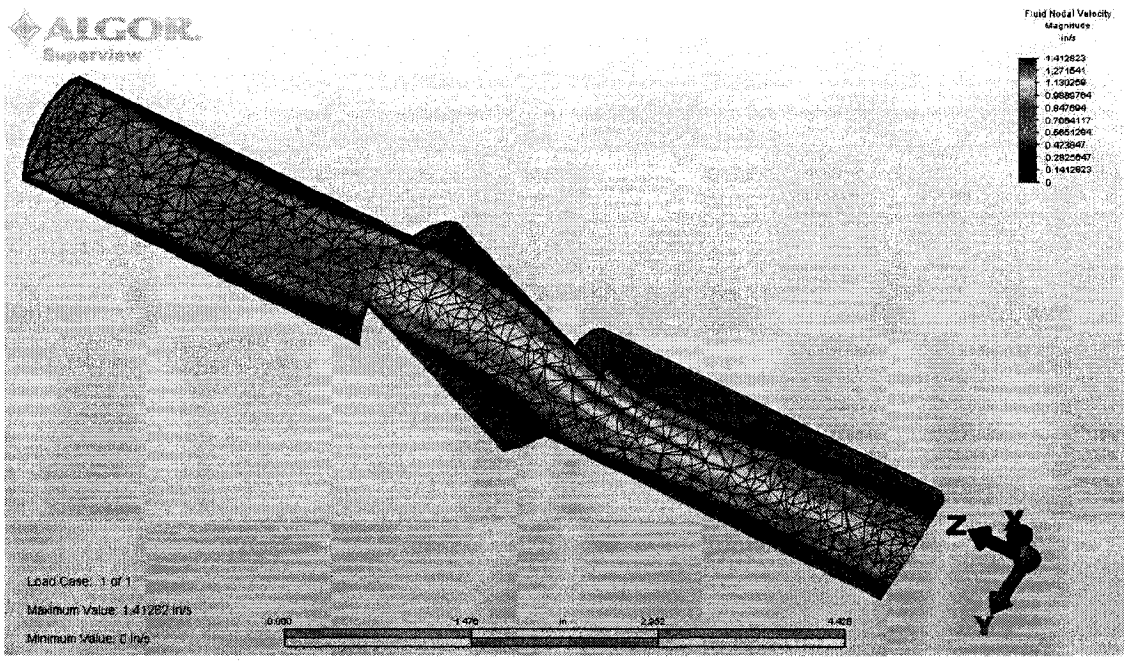


FIGURE 7.12 Velocity distribution (plot by magnitude)

general agreement with the anticipated results.

7.4.2 Unsteady flow

Figure 7.15 is the velocity results for the unsteady simulation plotted by magnitude. From this figure we can see that, the final fluid velocity distribution exhibits a similar behavior to the steady flow for a sufficiently long time.

Figure 7.16 illustrates the vertex pattern developed in the downstream area. Since all the nodes demonstrate the same convergent behavior, we only choose one node#2794 to illustrate the velocity varication, as shown in Fig. 7.17. Figure 7.18 illustrates the internal pressure distribution of this

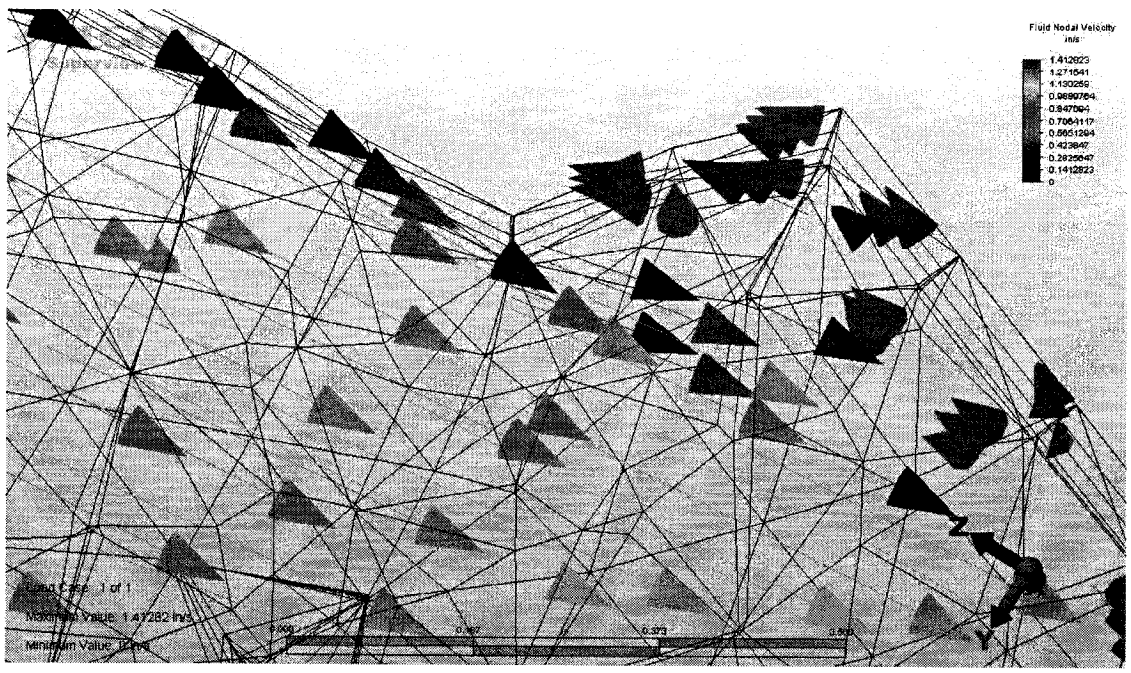


FIGURE 7.13 Vertex pattern

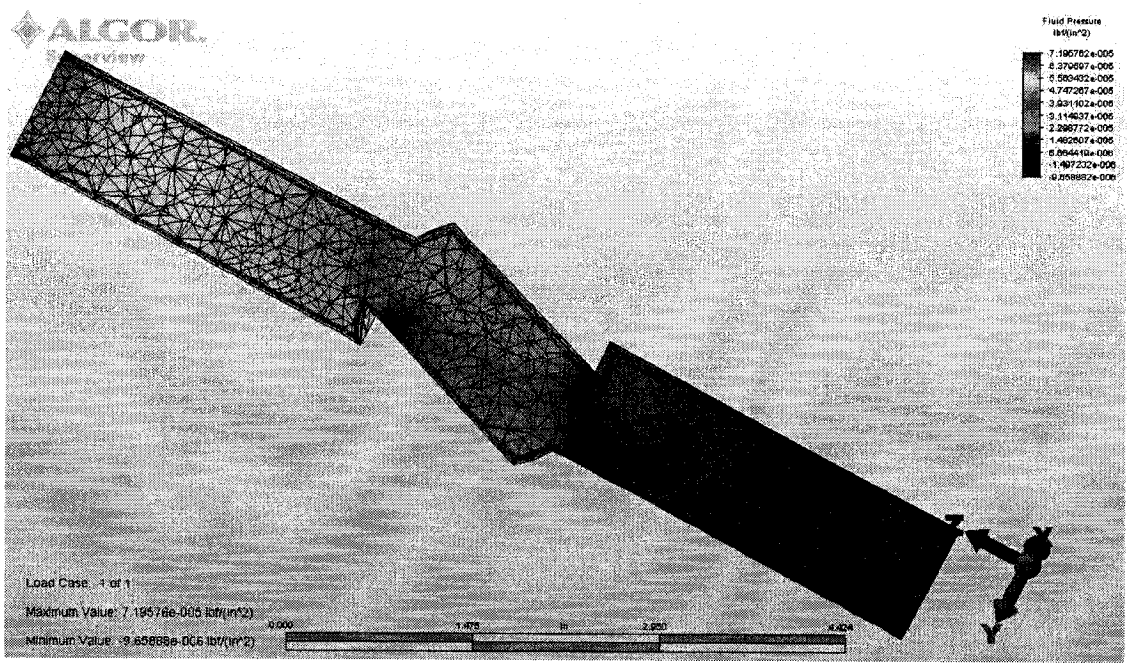


FIGURE 7.14 Pressure distribution

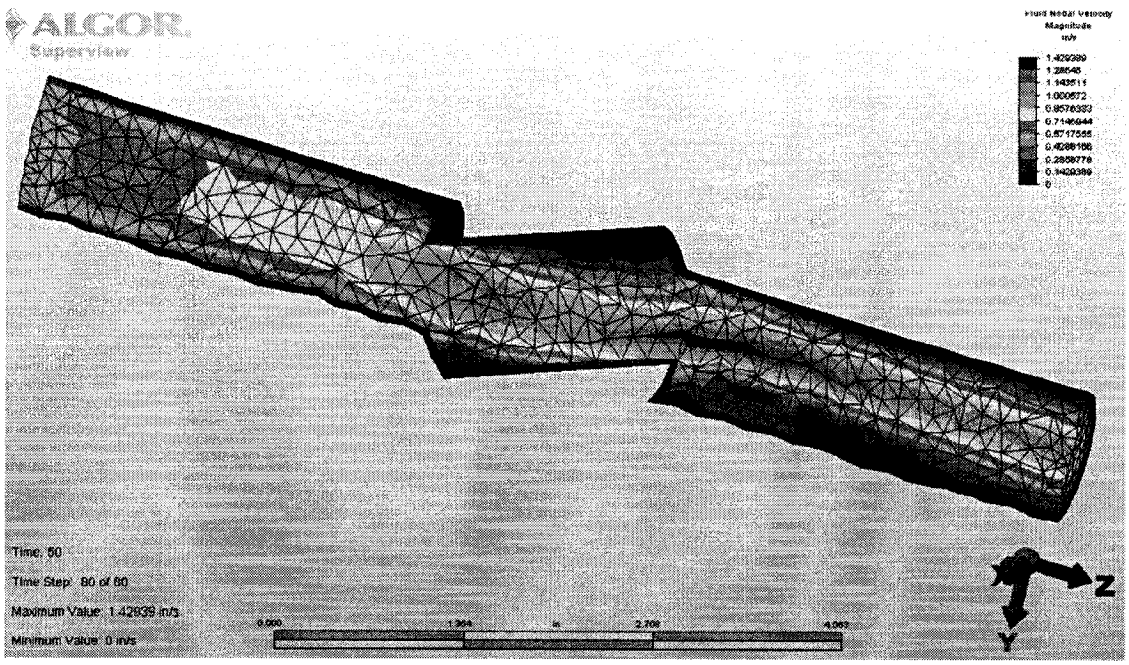


FIGURE 7.15 Velocity distribution (plot by magnitude)

model.

7.5 Discussion

In this Chapter, a fluid flow simulation in a three-dimensional (3D) model of a ball valve is used to validate the proposed hybrid grid methodology. This configuration has both convex and concave shapes in its body. The most difficult task in this test case is how to correctly calculate normal directions at the singularity points. At each singularity point, four surfaces pass through and share the singular point. The traditional normal calculation method uses the geometric information surrounding the point to be propagated, which brings ambiguous normal directions at singularity points for this model.

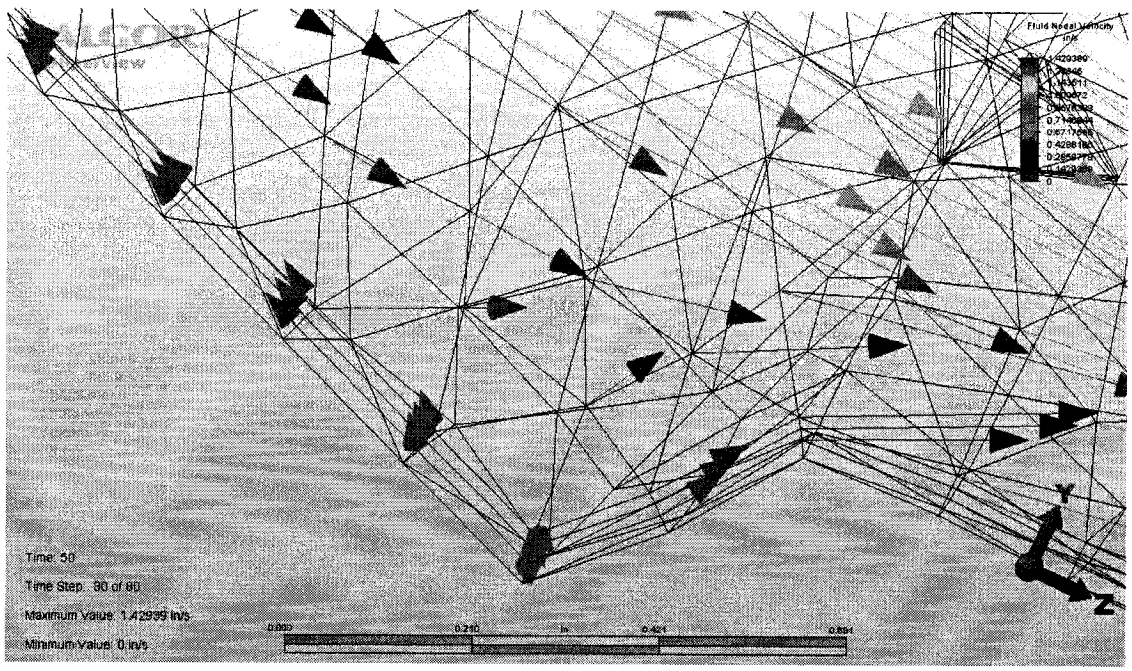


FIGURE 7.16 Vertex pattern

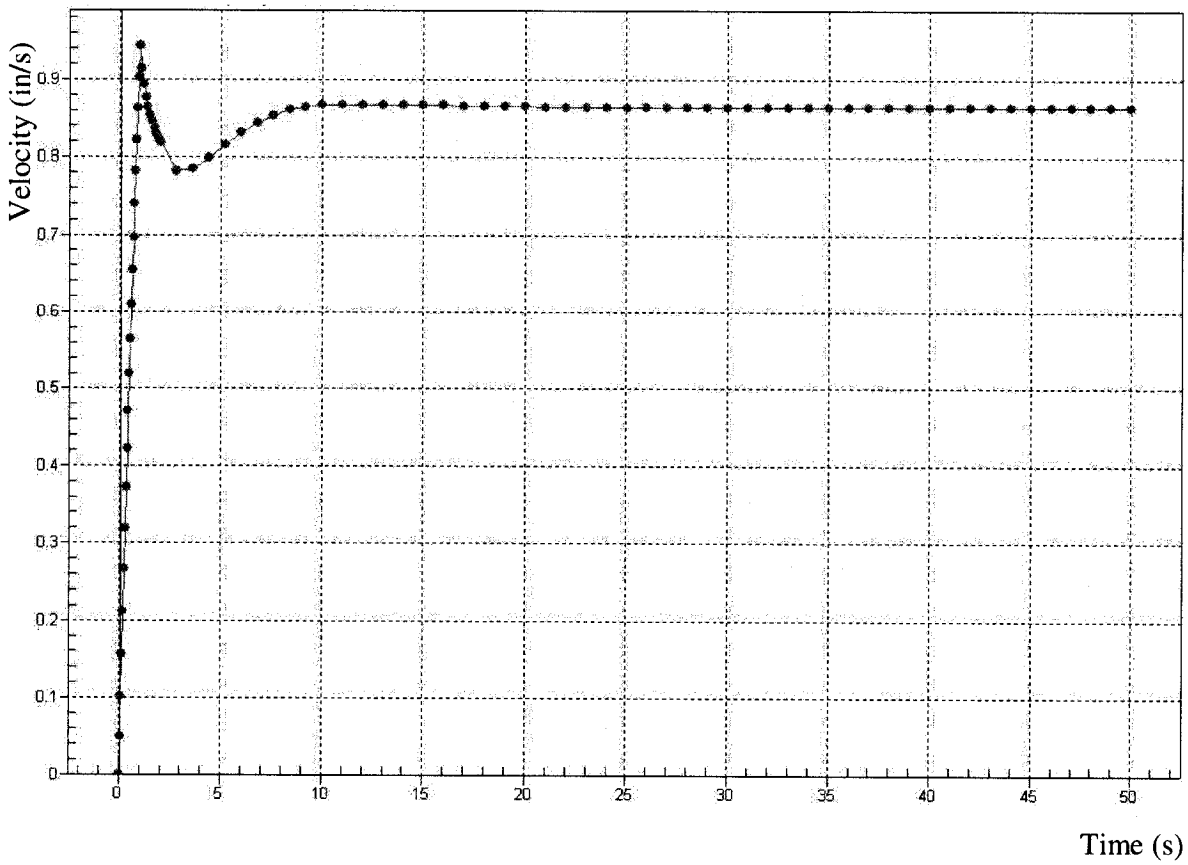


FIGURE 7.17 Velocity variation at node# 2794

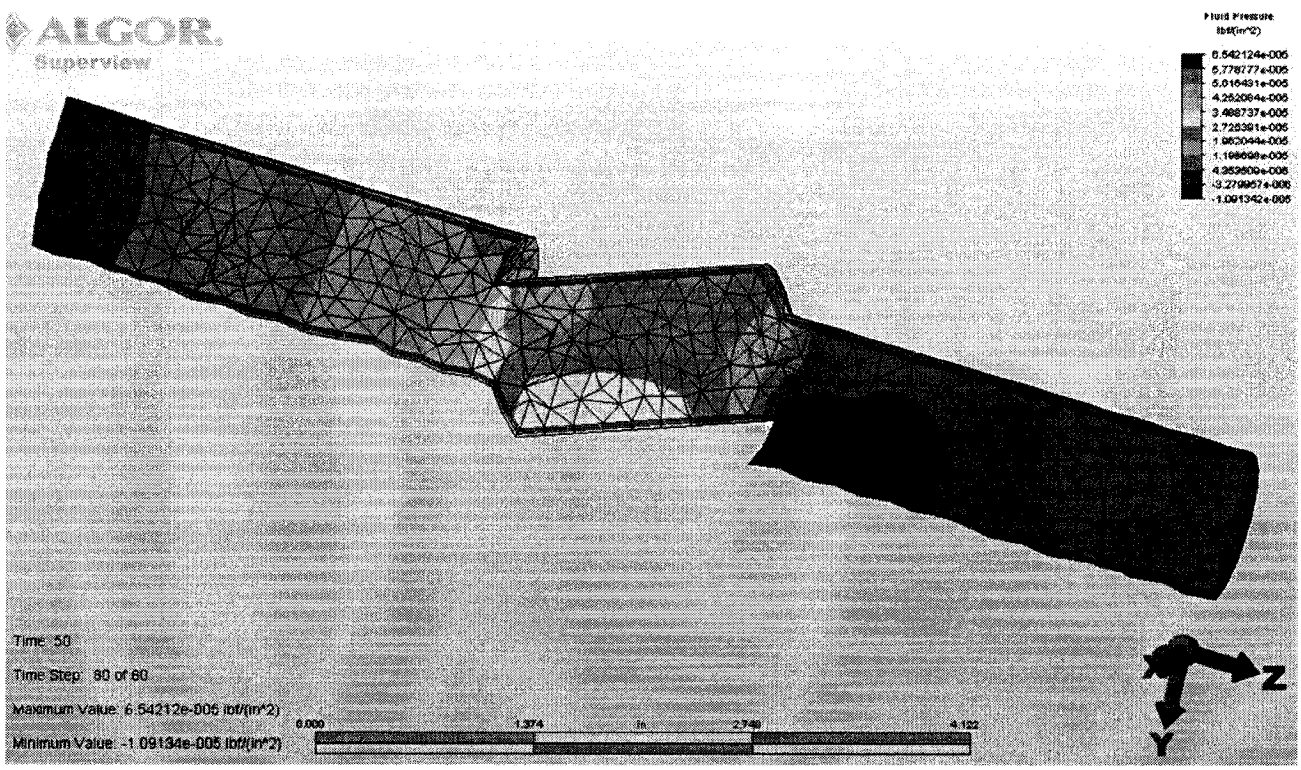


FIGURE 7.18 Pressure distribution

The boundary mesh is generated using the method proposed in Chapter 6, boundary mesh quality is measured from three aspects: 1, volume distribution 2, aspect ratio, and 3, Normalized equiangular skewness. These results show that the quality of the boundary mesh is acceptable. Also, this Chapter illustrated the sharp corner propagation behaviors and how the singularity problem is solved using the proposed normal calculation method. Finally, the fluid flow analysis under both steady and unsteady conditions are computed using commercial software to evaluate the vortex phenomenon. The results show that boundary mesh can correctly capture the vortex flow at the sharp corner area, and the result is converged after a certain period under the unsteady fluid conditions.

CHAPTER 8

CONCLUSION

8.1 Conclusions

The main contributions of this research achieved are classified into three types per the objectives addressed in Chapter 1. This section summarizes them accordingly.

8.1.1 Objective 1: Enhancement of the feature capturing capabilities

A new boundary mesh generation process for viscous flow is developed using a new front propagation technique. This method has several improvements over the previous front offset techniques:

- a new simplified mathematical model based on the Eikonal equation to solve a weak formulation of the front propagation problem. This mathematical model establishes a direct relationship between the points to propagate and the offset distance.
- this method intrinsically prevents self-intersections (global and local) by constructing a weak solution to the offset problem. In this method, corners and cusps are naturally handled.
- the formulation of the front propagation problem allows to propagate both convex and concave shapes using the same algorithm, and permeable boundaries such as inlet and outlet surfaces can be accounted for through the addition of special boundary conditions.

- applications to an arbitrary complex domain, i.e. a concave, or a convex shape, it may have sharp corners, or even be a multi-connected domain.

- the ability to handle the un-uniform front propagation applications by varying front propagation speed.

- the validation of the fast sweeping numerical scheme in the context of 3D applications with both smooth and discontinuous surfaces. As expected, the number of sweeps required to achieve field convergence depends only on the dimension of the computational domain, as 4 sweeps are needed in 2D and 8 in 3D, making the fast sweeping scheme extremely efficient numerically.

- preservation of the parametric information from the original front to the propagated front, i.e. topological connectivities of the original front.

- The description of the front can be continuous, ie, NURBS represented curves/surfaces, or non-continuous surface, ie, discretized line segments (2D) or triangulated faces (3D).

- The propagation techniques is quasi-linear first order to mesh size, the normal directions are dependent on the accuracy of the weak solution field only.

- The numerical scheme used for solving this PDE is a first order scheme. This research verifies that the bounding error is $|hlog(h)$ for an arbitrary set of discrete points, and the bounding error is $hlog(1/h)$ for a smooth surface.

- Normal direction at each front node to propagate is represented by the ϕ field rather than by the geometric information of each front node, i.e. the average of the surface normals attached to the node. The way of normal calculation ensures that the resulting propagation avoids self-intersections.

The boundary mesh generation method is validated using a 3D ball valve model. From the results of volume distribution, aspect ratio and normalized equiangular skewness, we can see that the quality of boundary mesh is acceptable, and it provides a convergent results for both steady and unsteady flow.

8.1.2 Objective 2: Increase of the level of automation

For very complicated geometrical models, manually quality checking is normally required when the traditional boundary layer mesh generation method is used. This is because sometimes the application may fail to generate a valid boundary mesh due to the self-intersections problems. The proposed front technique can successfully prevent both global and local self-intersection problems, there is no need to detect the self-intersection problems during propagation process, thus it improves the automation the boundary layer meshing.

- At viscous regions, the domain is automatically decomposed into face-matching multi-blocks while semi-structured grids are constructed, which makes the quality of stretched meshes are easily guaranteed, since optimization processes are carried out in each individual block. Also, mesh size is easily controlled by properly setting the speed function in the offset algorithm.
- The proposed The desired resulting mesh is a non-overlapped mesh because it avoids the interpolation as this is a source of inaccuracies.
- The physical domain can be an arbitrary complex domain. It may have both convex, and concave shape, a multiple-connected domain, or sharp corners, moreover, it can be extended to an open domain.

- It achieves a high level of automation. This algorithm works like a black-box, the input will be geometric and far-field boundaries, and topological information, the output will be the mesh points and connectivity. It does not require a human intervention during the mesh generation process.

8.1.3 Objective 3: Improvement of the computational efficiency

The utilization of the fast sweeping method for solving the Eikonal equation improves the computational complexity to $\Omega(M)$ compared with the fast marching method which computational complexity is $M\Omega(M)$. Also the sweeping property is suitable of using the parallelized algorithm which can further improve the sweeping efficiency. A primary parallized algorithm has been investigated, and it shows that the efficiency of the algorithm can be greatly improved when paralleled algorithm is applied.

8.2 Future works

There are several suggested future works being suggested to further improve the front propagation technique and the quality of the boundary layer mesh, which include:

1. One of the future research direction is to investigate the un-even space propagation equation. As discussed in the work of Qian *et al* (2006(1) and Qian *et al* (2006(2), a variable function can be used to represent the spatial dependent propagating speed. Moreover, anisotropic metric can be incorporated in the Eikonal equations.

2. A higher order numerical scheme for solving the distance offset equation is to be investigated to acquire more accurate solution of the proposed distance offset equation. A high order fast sweeping methods for eikonal equations has been presented in Zhang *et al* Zhang *et al* (2004), which can be used to improve the quality of the normal calculations. The usage of high order numerical scheme will provide a better propagation result especially for the offset-surface construction applications.

3. This thesis presents an optimization algorithm, which is discussed in Chapter 6, to optimize the boundary mesh. However, it is only limited to block-structured boundary mesh, and its tradeoff is to sacrifice the quality of skewness when improving the quality of volume distribution. The improvement of a more generalized optimization algorithm is to be investigated, especially when surface mesh is unstructured mesh.

4. One future work of this present method is to further study how to determine a proper value of dt used in the fourth-order Runge-Kutta method. If the value of dt is too small, then it will take a long time to reach the propagated distance; if the value of dt is too large, then the first propagation may exceed the specified propagation distance.

BIBLIOGRAPHY

BAKER, T. (1991). Single block mesh generation for a fuselage plus two lifting surfaces. *Proceedings of the International Conference on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields, 1991, p 261.*

BERGMAN, M. (1990). Development of numerical techniques for inviscid hypersonic flows around re-entry vehicles. *PhD thesis, INP Toulouse.*

BLOMGREN, R. (1981). B-spline curves, boeing document, class notes, b-7150-bb-wp-2811d-4412.

BONET, J. AND PERAIRE, J. (1991). An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems. *Int. J. Numer. Meth. Engng*, 31, 1–17.

H. BOROUCHAKI, HECHT, F., E. SALTEL AND P. L. GEORGE, Reasonably Efficient Delaunay Based Mesh Generator in 3 Dimensions, *Proceedings 4th International Meshing Roundtable, Sandia National Laboratories* 3–14

BOROUCHAKI, H. AND GEORGE, P.-L. (1996). Maillage de surfaces paramétiques. Partie II: Exemples d'applications. Technical report 2944, Institut National de Recherche en Informatique et en Automatique, France.

BOROUCHAKI, H., GEORGE, P.-L., HECHT, F., LAUG, P. AND SALTEL, É. (1997). Delaunay mesh generation governed by metric specification. Part I. Algorithms. *Finite Elem. in Anal. and Design*, 25, 61–83.

BOSSEN, F. J. AND HECKBERT, P. S. (1996). A pliant method for anisotropic mesh generetion,

fifth international meshing roundtable. *Fifth International Meshing Roundtable*. Sandia National Laboratories, Pittsburgh, Pennsylvania.

BOUDREAU, J., PIPERNI, P., MOKHTARIAN, F., KAFYEKE and F. (1997). Grid generation and euler calculations on the cf-18 aircraft. *In 6th Aerodynamics Symposium of the Canadian Aeronautics and Space institute.*

M. BOUE AND P. DUPUIS.(1999) Markov chain approximations for deterministic control problems with affine dynamics and quadratic costs in the control. *SIAM J. Numer. Anal.*, 36 667–695

CASTRO-DíAZ, M. J., HECHT, F., MOHAMMADI, B. AND PIRONNEAU, O. (1997). Anisotropic unstructured mesh adaptation for flow simulation. *Int. J. Numer. Meth. Fluids*, 25, 475–491.

CAUGHEY, D. and JAMESON, A. (1980). Progress in finte-volume calculations for wing fuselage combinations. *AIAA Journal, v18, n11, p1281-1288.*

CONNELL, S. D. AND BRAATEN, M. E. (1995). Semistructured mesh generation for three dimensional Navier-Stokes calculations. *AIAA Journal*, 33, 1017–1024.

COQUILLART, S. (1987). Computing offsets of b-spline curves. *Comp.-Aided Design*, 19, 305–309.

CORDOVA, J. (1992). Towards a theory of automated elliptic mesh generation. *In NASA Workshop on Software Systems for Surface Modeling and Grid Generation, NASA Conference Publication, n 3143, p231.*

J. DELLINGER AND W. W. SYMES.(1997) Anisotropic finite-difference traveltimes using a Hamilton-Jacobi solver. In 67th Ann. *Internat. Mtg., Soc. Expl. Geophys., Ex- panded Abstracts*, pages 1786–1789. Soc. Expl. Geophys.

ENGQUIST, B. AND OSHER, S. (1980). Stable and entropy-satisfying approximations for transonic flow calculations. *Math. Comp., Vol.34(149) P45-75.*

M. FALCONE AND R. FERRETTI.(1994). Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations. *Numer. Math., Vol.67 315–344*

FARIN, G. (1997). Curves and Surfaces for Computer Aided Geometric Design. *Academic Press*, New York.

FAROUKI, R. T. (1986). The approximation of non-degenerate offset surfaces. *Comp.-Aided Geom. Design*, 3, 15–43.

FORTIN, A. AND GARON, A. (2000). *Les éléments finis: de la théorie à la pratique.* École Polytechnique de Montréal.

GARIMELLA, R. (1998). *ANISOTROPIC TETRAHEDRAL MESH GENERATION.* PhD thesis, Faculty of Rensselaer Polytechnic Institute.

GARIMELLA, R. AND SHEPHARD, M. (2000). Boundary layer mesh generation for viscous flow simulations. *Int. J. Numer. Meth. Engng*, 49, 193–218.

P.L. GEORGE, F. HECHT ,AND E. SALTEL. (1991). Automatic Mesh Generator with Specified Boundary, *Computer Methods in Applied Mechanics and Engineering, North-Holland, Vol 92* 269–288.

GHIA, U., GHIA, K. and RAMAMURTI, R. (1983). Hybrid c-h for turbomachinery cascades. *American Society of Mechanical Engineers, Fluids Engineering Division (Publication) FED, v 5, p143-149.*

GLIMM, J., GROVE, J., LI, X. AND TAN, D. (2000). Robust computational algorithms for dynamic interface tracking in three dimensions. *SIAM J. Sci. Comp.*, 21, 2240–2256.

M. G. CRANDALL AND P. L. LIONS. (1984) Two approximations of solutions of Hamilton-Jacobi equations. *Math. Comp.* 43, 119.

GUIBAULT, F. (1998). *Un mailleur hybride structuré/non structuré en trois dimensions.* PhD thesis, École Polytechnique de Montréal.

HASSAN, O., MORGAN, K., PROBERT, E. AND PERAIRE, J. (1996). Unstructured tetrahedral mesh generation for three-dimensional viscous flows. *Int. J. Numer. Meth. Engng*, 39, 549–567.

HASSAN, O., PROBERT, E., MORGAN, K. AND PERAIRE, J. (1995). Mesh generation and adaptivity for the solution of compressible viscous high speed flows. *Int. J. Numer. Meth. Engng*, 38, 1123–1148.

HELMSEN, J. (1994). A comparison of three-dimensional photolithography development methods. *Ph.D. Dissertation, EECS, University of California, Berkeley.*

J. HELMSEN, E. PUCKETT, P. COLELLA, AND M. DORR. (1996). Two new methods for simulating photolithography development in 3-D. *Proc. SPIE* 2726 253–261.

HOFFMAN, J. (1992). *Numerical Methods for Engineers and Scientists.* McGraw-Hill Inc.

JAMESON, A. (1974). Iterative solution of transonic flows over airfoils and wings, including flows at mach 1. *Comm. Pure. Appl. Math., v27, p283-309.*

KALLINDERIS, Y. (1995). Adaptive hybrid prismatic/tetrahedral grids. *Int. J. Numer. Meth. Fluids*, 20, 1023–1037.

KALLINDERIS, Y., KHAWAJA, A. AND MCMORRIS, H. (1995). Hybrid prismatic/tetrahedral grid generation for complex geometries. *33rd Aerospace Sciences Meeting and Exhibit.* No. AIAA-95-0211.

C.Y. KAO, S.J. OSHER, AND J. QIAN.(2004) Lax-Friedrichs sweeping schemes for static Hamilton-Jacobi equations. *J. Comp. Phys.* 196 367–391.

S. KIM AND R. COOK. 3-D traveltime computation using second-order ENO scheme. *Geophysics, v64, 1867–1876.*

KIMMEL, R. AND BRUCKSTEIN, A. M. (1993). Shape offsets via level sets. *Comp.-Aided Design*, 25, 154–162.

KIM, B. and EBERHARDT, S. (1995). Automatic multi-block grid generation for high-lift configuration wings. *Proceedings of the Surface Modeling, Grid Generation and Related Issues in Computational Fluid Dynamics Workshop, NASA Conference Publication 3291, p. 671, NASA Lewis Research Center, Cleveland, OH, May.*

R. KIMMEL AND J. A. SETHIAN. (1998). Computing geodesic paths on manifolds. *Proc. Natl. Acad. Sci. USA* 95 8431–8435.

KULCZYCKA, A. AND NACHMAN, L. (2002). Qualitative and quantitative comparisons of b-spline offset surface approximation methods. *Comp.-Aided Design*, 34, 19–26.

KUMAR, R. G. V. V., SHASTRY, K. G. AND PRAKASH, B. G. (2003). Computing offsets of trimmed nurbs surfaces. *Comp.-Aided Design*, 35, 411–420.

LEE, E. (2003). Contour offset approach to spiral toolpath generation with constant scallop height. *Comp.-Aided Design*, 35, 511–518.

LEVEQUE, R. (2002). Finite volume methods for hyperbolic problems. *Cambridge University Press, ISBN 0 521 81087 6 hardback.*

LIPSKI, W., LODI, E., LUCCIO, F., MUGNAI, C. and PAGLI, L. (1979). On two-dimensional data organization ii. *Annales Societis Mathematicae Polonae, Series IV: Fundamentae Informaticae II, p245-260.*

LODI, E., LUCCIO, F., MUGNAI, C. and PAGLI, L. (1979). On two-dimensional data organization i. *Annales Societis Mathematicae Polonae, Series IV: Fundamentae Informaticae II, p211-226.*

LÖHNER, R. (1993). Matching semi-structured and unstructured grids for Navier-Stokes calculations. *11th AIAA Computational Fluid Dynamics Conference*. AIAA, Orlando, FL, no. AIAA–93–3348–CP, 555–564.

LÖHNER, R. AND CEBRAL, J. (2000). Generation of non-isotropic unstructured grids via directional enrichment. *Int. J. Numer. Meth. Engng*, 49, 219–232.

MALLADI, R. AND SETHIAN, J. A. (1996). Level set and fast marching methods in image processing and computer vision. *IEEE International Conference on Image Processing*, 1, 489–492.

MANHARDT, P. and BAKER, A. (1982). Automated three-dimensional grid refinement on a minicomputer. *In Numerical Grid Generation.*

MARCHANT, M. AND WEATHERILL, N. P. (1994). Unstructured grid generation for viscous flow simulations. *Proc. 4th Int. Conf. on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*. Pineridge Press, Swansea, UK.

MARCHANT, M. J., WEATHERILL, N. P. AND HASSAN, O. (1997). Adaptation of unstructured

grids for transonic viscous flow simulation. *Finite Elements in Analysis and Design*, 25, 199–218.

MARCUM, D. L. AND WEATHERILL, N. P. (1995). Unstructured grid generation using iterative point insersion and local reconnection. *AIAA J.*, 33, 1619–1625.

MAVRIPLIS, D. J. (1990). Adaptive mesh generation for viscous flows using Delaunay triangulation. *J. Comp. Phys.*, 90, 271–291.

NAKAHASHI, K. AND SHAROV, D. (1995). Direct surface triangulation using the advancing front method. *Proc. 12th AIAA CFD Conference.*

NAKAHASHI, K., SHAROV, D., KANO, S. AND KODERA, M. (1999). Application of unstructured hybrid grid method to high-reynold number viscous flows. *Int. J. Numer. Meth. Fluids*, 31, 97–111.

OSHER, S. (2003). *Geometric Level Set Methods in Imaging, Vision, and Graphics.* Springer-Verlag New York, Inc.

OSHER, S. AND SETHIAN, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *J. Comp. Phys.*, 79, 12–49.

PARK, S. C. AND CHUNG, Y. C. (2003). Mitered offset for profile machining. *Comp.-Aided Design*, 35, 501–505.

PIEGL, L. A. AND TILLER, W. (1999). Computing offsets of nurbs curves and surfaces. *Comp.-Aided Design*, 31, 147–156.

PIEGL, L. A. AND TILLER, W. (1995). The NURBS Book. *Springer-Verlag.*

PIPERNI, P., MOKHTARIAN, F. and KAFYEKE, F. (1992). Multi-block grid generation for complete aircraft configurations. *Canadian Aeronautics and Space Journal, 38(4).*

PIPERNI, P. (1994). Multi-block grid generation with cad-based domain decomposition techniques. *In Numerical Grid Generation in Computational Fluid Dynamics and Related Fields, p109-121, Swansea, U.K. Pineridge Press Ltd.*

PIPERNI, P. (2003). *A Fundamental Solution to the Problem of Domain Decomposition in Structured Grid Generation.* Thèse de doctorat, Génie Mécanique, École Polytechnique de Montréal.

PIPERNI, P. and CAMARERO, R. (2003). A fundamental solution to the problem of domain decomposition in structued grid generation. *AIAA 2003-0951, 41st Aerospace Sciences Meeting & Exhibit, 6-9 January.*

PIRZADEH, S. (1993). Unstructured viscous grid generation by advancing-layers method. *AIAA 11th Applied Aerodynamics Conference.* Monterey, CA, no. AIAA-93-3453, 420–434.

PIRZADEH, S. (1994). Viscous unstructured three-dimensional grids by the advancing-layers method. *32nd AIAA Aerospace Sciences Meeting.* Reno, NV, no. AIAA-94-0417.

PIRZADEH, S. (1996). Three-dimensional unstructured viscous grids by the advancing-layers method. *AIAA J.*, 34, 43–49.

J. QIAN, Y. ZHANG, H. ZHAO. (2006) Fast sweeping methods for Eikonal equations on triangulated meshes. *SIAM Journal on Numerical Analysis, to appear. UCLA CAM report:05-07* http://www.math.ucla.edu/applied/cam

J. QIAN, Y. ZHANG, H. ZHAO. (2006) A Fast sweeping method for static convex Hamilton-Jacobi equations. *UCLA CAM report:06-37* http://www.math.ucla.edu/applied/cam/

J. QIAN AND W. W. SYMES (2002). Adaptive finite difference method for traveltime and amplitude. *Geophysics*, 67 167–176, 2002.

F. QIN, Y. LUO, K. B. OLSEN, W. CAI, AND G. T. SCHUSTER. (1992) Finite difference solution of the eikonal equation along expanding wavefronts. *Geophysics* V.57 P.478–487

ROBERTS, A. (1982). Automatic topology generation and generalized b-spline mapping. *In Numerical Grid Generation.*

ROUY, E., AND TOURIN, A. (1992). A viscosity solutions approach to shape-from-shading. *SIAM J. Num. Anal.*. 29(3), 867–884.

RUPPERT, P. and LEE, K. (1980). Transonic flow computations using grid systems with block structure. *In 7th International Conference on Numerical Methods in Fluid Dynamics.*

W. A. JR. SCHNEIDER, K. RANZINGER, A. BALCH, AND C. KRUSE. A dynamic programming approach to first arrival traveltime computation in media with arbitrarily distributed velocities. *Geophysics*, 57:39–50.

SCHONFELD, T. and WEINERFELT, P. (1991). The automatic generation of quadrilateral multiblock grids by the advancing front technique. *In Numerical Grid Generatin in Computational Fluid Dynamics, A.S. Arcilla, J. Hauser, P.R. Eiseman, and J.F. Thompson (Eds.), p743, Proceedings of the 3rd International Grid Conference, North Holland, Barcelona, Spain, June.*

SCHUSTER, D. and ATTA, E. (1989). Flight loads prediction methods for fighter aircraft. *Volume II-Complete Aircraft Mesh Program (CAMP) User's Guide. Technical Report, WRDC.*

SCHUSTER, D. (1992). Batch mode grid generation: An endangered species? *In Proceedings of the Software Systems for Surface Modeling and Grid Generation Workshop, R.E. Smith (Ed.), NASA Conference Publication 3143, p487, NASA Langley Research Center, Hampton, VA.*

SETHIAN, J. A. (1982). *An Analysis of Flame Propagation.* Ph.d. dissertation, University of California, Berkeley, CA.

SETHIAN, J. A. (1985). Curvature and the evolution of fronts. *Comm. in Math. Phys.*, 101, 487–499.

SETHIAN, J. A. (1988). The design of algorithms for hypersurfaces moving with curvature-dependent speed. *Nonlinear Hyperbolic Equations-Theory, Numerical Methods, and Applications, Notes on Numerical Fluid Mechanics.* Ballman, J. and Jeltsch, R. Eds., Vieweg, vol. 24.

SETHIAN, J. A. (1996). A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences of the United States of America*, 93, 1591–1595.

SETHIAN, J. A. (1999). *Level set methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science.* Cambridge University Press.

SHAROV, D., LUO, H., BAUM, J. AND LÖHNER, R. (2003). Unstructured navier-stokes grid generation at corners and ridges. *International Journal for Numerical Methods in Fluids*, 43, 717–728.

SHAROV, D. AND NAKAHASHI, K. (1998). Hybrid prismatic/tetrahedral grid generation for viscous flow applications. *AIAA J.*, 36, 157–162.

SHAW, J., GEORGALA, J. and WEATHERILL, N. (1988). The construction of component adaptive grids for aerodynamic geometries. *In Numerical Grid Generation in Computational Fluid Dynamics, '88, p383–394. Pineridge Press.*

SHAW, J. and WEATHERILL, N. (1992). Automatic topology generation for multiblock grids. *Applied Mathematics and Computation, Volume 52, Issues 2-3, December, p355-388.*

SMEREKA, P. AND SETHIAN, J. (2003). Level set methods for fluid interfaces. *Annual Review of Fluid Mechanics*, 35, 341–372.

SORENSON, R. and STEGER, J. (1983). Grid generation in three dimensions by poisson's equation in a visually interactive networking environment. *In Proceedings from the NASA Workshop on Software Systems for Surface Modeling and Grid Generation.*

SORENSON, R. (1988). Three-dimensional zonal grids about arbitrary shapes by poison's equation. *In Numerical Grid Generation in Computational Fluid Dynamics.*

SPEKREIJSE, S. P. (1995). Elliptic grid generation based on laplace equations and algebraic transformations. *J. Comp. Phys.*, 118, 38–61.

STEWART, M. (1992a). Domain-decomposition algorithm applied to multi-element airfoil grids. *AIAA Journal, Vol. 30, No. 6, p1457-1461, June.*

STEWART, M. (1992b). A multiblock grid generation technique applied to a jet engine configuration. *Proceedings of the Software Systems for Surface Modeling and Grid Generation Workshop, R.E. Smith (Ed.), NASA Conference Publication n.3143, p477, NASA Langley Research Center, Hampton, VA.*

SULLIVAN, J. AND ZHANG, J. (1997). Adaptive mesh generation using a normal offsetting technique. *Finite Elements in Analysis and Design*, 25, 275–295.

SUN, Y. F., NEE, A. Y. C. M. AND LEE, K. S. (2004). Modifying free-formed nurbs curves and surfaces for offsetting without local self-intersection. *Comp.-Aided Design*, 36, 1161–1169.

FLUENT INC. (2003). TGrid 3.5 User's Manual . April, 2003.

THERMEXCEL (2003). http://www.thermexcel.com/english/tables. *June 2003.*

THOMPSON, J. (1987). A composite grid generation code for general 3d regions. *In AIAA conference, AIAA-87-0275, 25th AIAA Aerospace Sciences Meetings, Reno, NV, January.*

THOMPSON, D. S., CHALASANI, S. AND SONI, B. K. (2000). Smoothing of volume meshes generated by extrusion from surface meshes of arbitrary topology. *Proceedings of the 9th International Meshing Roundtable.*

TILLER, W. AND HANSON, E. (1984). Offsets of two-dimensional profiles. *IEEE Computer Graphics and Applications*, 4, 36–46.

J. VAN TRIER AND W. W. SYMES. Upwind finite-difference calculation of traveltimes. *Geophysics*, 56:812821

TSAI, Y. R., CHENG, L. T., OSHER, S. AND ZHAO, H. K. (2003). Fast sweeping algorithms for a class of hamilton-jacobi equations. *SIAM Journal on Numerical Analysis*, 41, 673–694.

R. TSAI, L.-T. CHENG, S. J. OSHER, AND H. K. ZHAO (2003). Fast sweeping method for a class of Hamilton-Jacobi equations. *SIAM J. Numer. Anal.*, 41 673–694.

J. N. TSITSIKLIS. (1995) Efficient algorithms for globally optimal trajectories. *IEEE Tran. Automatic Control* 40 1528–1538.

VALLET, M.-G., HECHT, F. AND MANTEL, B. (1991). Anisotropic control of mesh generation based upon a Voronoi type method. A. S. Arcilla, J. Häuser, P. R. Eiseman and J. F. Thompson, Editors, *Third International Conference on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*. North-Holland, Barcelona, Spain, 93–103.

J. VIDALE. Finite-difference calculation of travel times. *Bull. Seis. Soc. Am.*, 78:20622076

WANG, Y.-L., GUIBAULT, F. AND CAMARERO, R. (September 2005). Automatic near-body domain decomposition using the eikonal equation. *Proceedings of the 14th International Meshing Roundtable*. Sandia National Laboratory.

WANG, Y.-L., MURGIE, S. (September 2006). Hybrid Mesh Generation For Viscous Flow Simulations. *Proceedings of the 15th International Meshing Roundtable*. Sandia National Laboratory.

WANG, Y.-L., GUIBAULT, F. AND CAMARERO, R. (to be published). Eikonal equation-based Front Propagation for Arbitrary Complex Configurations. *Int. J. Numer. Meth. Engng*.

WANG, Y.-L., GUIBAULT, F., CAMARERO, R. AND TCHON, K.-F. (June 2005). A parametrization transporting surface offset construction method based on the eikonal equation. *17th AIAA CFD Conf.*

WEATHERILL, N. and FORSEY, C. (1984). Grid generation and flow calculations for complex aircraft geometries using a multi-block scheme. *In AIAA Conference, AIAA-84-1665, AIAA 17th Fluid Dynamics, Plasma Dynamics, and Laser Conference, Snowmass, CO.*

Y. T. ZHANG, H. K. ZHAO, AND J. QIAN. High order fast sweeping methods for static Hamilton-Jacobi equations. *UCLA CAM; J. Sci. Comp. (under revision)* 4–37.

H.K. ZHAO, S. OSHER, B. MERRIMAN, AND M. KANG. (2000) Implicit and non-parametric shape reconstruction from unorganized points using variational level set method. *Computer Vision and Image Understanding*. 80 295–319.

Y. T. ZHANG, H. K. ZHAO, AND J. QIAN. High order fast sweeping methods for Eikonal equations. *in SEG 74, 2004* 1901–1904.

ZHAO, H. K. (2005). A fast sweeping method for eikonal equations. *Mathematics of Computation*, 74, 603–627.