한국 CAD/CAM 학회 논문집
제 12권 제 3호 2007년 6월 pp. 220-232

학술논문

R-function을 이용한 형상의 음함수 모델링 및 해석

신헌주*, 김태완**, 신동우***

Geometric Implicit Function Modeling and Analysis Using R-functions

Heonju Shin*, Tae-wan Kim** and Dongwoo Sheen***

ABSTRACT

Current geometric modeling and analysis are commonly based on B-Rep modeling and a finite elements method respectively. Furthermore, it is difficult to represent an object whose material property is heterogeneous using the B-Rep method because the B-Rep is basically used for homogeneous models. In addition, meshes are required to analyze a property of a model when the finite elements method is applied. However, the process of generating meshes from B-Rep is cumbersome and sometimes difficult especially when the model is deformed as time goes by because the topology of deforming meshes are changed. To overcome those problems in modeling and analysis including homogeneous and heterogeneous materials, we suggest a unified modeling and analysis method based on implicit representation of the model using R-function which is suggested by Rvachev. For implicit modeling of an object a distance field is approximated and blended for a complex object. Using the implicit function mesh-free analysis is possible where meshes are not necessary. Generally mesh-free analysis requires heavy computational cost compared to a finite elements method. To improve the computing time of function evaluation, we utilize GPU programming. Finally, we give an example of a simple pipe design problem and show modeling and analysis process using our unified modeling and analysis method.

Key words: Approximated distance implicit function, R-function, heterogeneous material modeling, mesh-free analysis, GPU programming

1. 서 론

현재까지 모델링과 해석 프레임워크는 대부분 B-Rep(boundary representation)모델링과 유한요소법을 이용하여 구성된다. 이러한 프레임워크에서 B-Rep 모델링은 이질속성물체를 표현하기 힘들다는 한계점을 갖는다[17,18]. 또한 유한요소법은 해석을 위해 B-Rep 모델의 형상정보를 그대로 이용하지 못하고 해석용 메쉬를 생성해야 한다는 한계점을 갖고 있다. 유한요소법에서 이러한 한계점은 해석결과를 다시 피드백(feedback)하여 모델을 수정할 때에 또 모델변환을 해야 하는 문제와 시간에 따라 움직이는 형상을 해석할 경우 계속 remeshing해야 하는 문제를 야기한다.

거리기반 음함수 모델링(approximated distance implicit function modeling)과 무요소법(mesh-free analysis)은 기존의 이러한 한계점을 해결할 수 있는 하나의 방법을 제시한다. 거리기반 음함수 모델링은 B-Rep모델링에 비하여 보다 쉽게 이질속성물체를 표현할 수 있다. 이러한 이질속성물체 모델링은 앞으로 수요가 더욱 증가할 것이다. 또한, 무요소법은 해석용 메쉬를 만들지 않고 거리기반 음함수 모델을 바로 해석 가능하다. 그러므로 유한요소법이 갖고 있던 메쉬 생성으로 인한 문제가 없으며 모델링과 해석을 통합할 수 있다. 그러나 무요소법은 유한요소법에 비하여 해석과정에서 많은 계산이 필요하다. 본 연구에서는 이러한 문제를 해결하고자 GPU 프로그래밍 기법을 무요소법에 적용하였다. Fig. 1은 본 연구의 통합된 모델링과 해석 프레임워크를 나타낸다.

Rvachev는 R-function과 R-function을 이용한 음함수 모델의 불리언 연산 등을 정리하였다[1,3,19]. 또한

*(주)아이너스기술
**교신저자, 종신회원, 서울대학교 조선해양공학과 및 해양시스템공학연구소
***서울대학교 수리과학부
- 논문투고일: 2006. 03. 27
- 심사완료일: 2007. 05. 29

Shapiro *et al.*은 여러 R-function 시스템을 이용하여 음함수 모델을 불리언하였을 때 각각의 미분성질에 대하여 연구하였다[8]. Biswas *et al.*은 거리기반 음함수를 이용하여 근사된 거리장(distance field)를 나타내는 방법을 연구하였으며 거리기반 음함수를 응용하여 이질속성물체를 모델링 하였다[13,14]. Rvachev는 곡선 위에 정의된 함수값들을 보간할 수 있는 transfinite interpolation을 연구하였고 이는 이질속성물체 모델링에 이용된다[11]. Rvachev *et al.*은 R-function 불리언을 이용하여 만들어진 음함수를 이용하여 solution structure를 정의하였다[10]. Tsukanov *et al.*은 solution structure를 이용한 무요소법을 제시하였다[12]. 최근 본래 그래픽 처리용으로 사용되는 GPU를 유한요소법, 수치해석 등과 같은 다른 범용으로 사용하고자 하는 많은 연구가 이루어지고 있다[15,16].

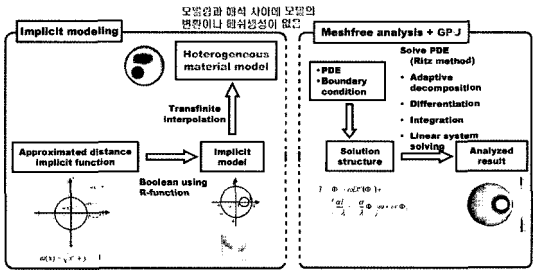

Fig. 1. 거리기반 음함수 모델링과 무요소해석 프레임워크.

2. 형상 모델링

2.1 기본형상 모델링

2.1.1 음함수 표현법

음함수 표현법은 표현하고자 하는 곡선이나 곡면을 음함수식 을 이용하여 표현한다[1,3]. 즉 주어진 음함수식을 만족하는 점들의 집합이 곧 표현하고자 하는 형상이 된다. Fig. 2는 단위 원 과 구를 음함수로 표현한 예이다. 이러한 음함수 표현법은 매개변수 표현법과 비교하여 다음과 같은 장점을 갖는다. 첫째, 임의의 위치에서 물체의 내외부를 쉽게 판단할 수 있다. 둘째, 불리언(Boolean) 연산이 쉽다. 보통 모델링 과정에서 복잡한 형상은 간단한 기본형상들을 서로 조합하여 만들어 진다. 불리언 연산이란 기본형상들을 서로 조합할 때 이용되는 논리연산을 의미한다. 기본형상이 매개변수식으로 표현된 경우 불리언 할 때 서로 교차계산(surface-surface intersection)을 수행해야 하지만 음함수로 표현된 경우 R-function을 이용한 간단한 함수계산 만으로 불리언 가능하다. 셋째, 곡선, 곡면, 기본형상들 간의 연결관계(topology)를 저장하지 않는다. 매개변수식을 이용한 모델링의 경우 매개변수 곡선, 곡면, 기본형상들의 연결관계를 half-edge, winged edge와 같은 자료구조에 저장해야 한다. 음함수를 이용한 모델링은 최종 모델이 하나의 음함수로 표현되므로 연결관계를 따로 정의할 필요가 없다.

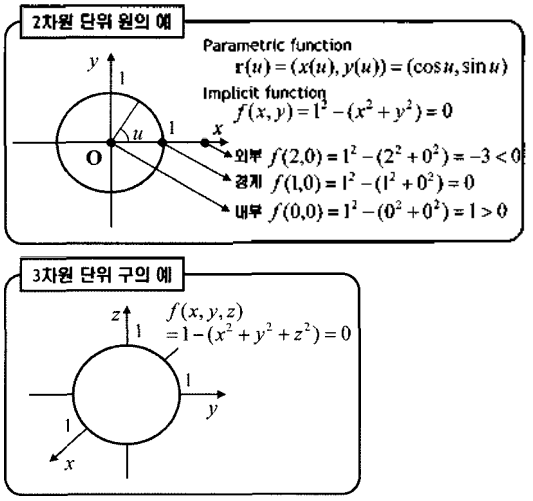

Fig. 2. 단위 원 및 구에 대한 음함수 예.

2.1.2 거리기반 음함수 표현법

기존의 음함수는 함수값이 0인 점들의 집합이 물체의 경계를 나타내고 양수, 음수인 점들의 집합은 각각 물체의 내외부를 나타낸다. 즉 함수값의 부호로 물체의 경계, 내부, 외부를 판단하지만 함수값의 부호 이외에 값 자체는 큰 의미가 없다. 거리기반 음함수란 함수값의 절대치가 물체의 경계로부터 함수값을 계산한 점까지의 최단거리를 계산하여 준다[13]. Fig. 3은 단위 원을 일반적인 음함수와 거리기반 음함수로 각각 표현한 예이다. 거리기반 음함수로 기본형상을 표

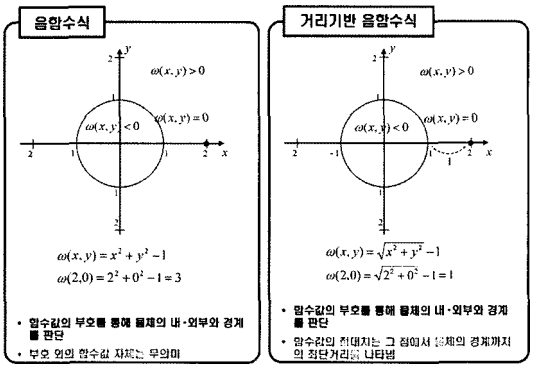

Fig. 3. 음함수와 거리기반 음함수 비교.

현하면 이질속성물체를 쉽게 표현할 수 있는 장점이 있다. 본 논문에서 제안하는 통합된 모델링과 해석 프레임워크에서는 모든 기본형상들을 거리기반 음함수로 표현한다.

2.1.3 Normalization condition

Fig. 3의 단위 원 예제에서와 같이 거리기반 음함수식와 일반적인 음함수는 함수의 형태가 다르다. 즉, 임의의 음함수가 거리기반 음함수가 되기 위해서는 특정한 조건을 만족해야만 한다. 이러한 조건을 normalization condition이라 한다[13]. Normalization condition은 다음과 같다.

$$1.\ \omega_{|\partial\Omega} = 0$$

$$2.\ \frac{\partial\omega}{\partial\mathbf{v}_{\partial\Omega}} = 1,\ \mathbf{v} = \text{normal to the boundary } \partial\Omega \quad (1)$$

$$3.\ \frac{\partial^k\omega}{\partial\overset{k}{\mathbf{v}}_{\partial\Omega}} = 0,\ k = 2,3,4,\ldots,m$$

식 (1)에서 첫번째 조건은 음함수값이 0인 점들의 집합이 물체의 경계를 나타낸다는 조건이다. 즉, 물체의 경계에서의 음함수값은 모두 0이다. 이는 음함수로 모델링하기 위해서 기본적으로 만족해야 하는 조건이다. 두번째 조건은 물체의 경계에 존재하는 임의의 점에서 법선벡터 방향으로의 한번 미분한 값이 1이 되어야 한다는 조건이다. 세번째 조건은 물체의 경계에 존재하는 임의의 점에서 법선벡터 방향으로 2번부터 k번까지 미분한 값이 모두 0이라는 조건이다. $k=m$일 때 음함수 ω는 m-th order로 normalization된다. Normalization 차수가 높을수록 보다 잘 근사된 거리기반 음함수가 된다. 만약 무한대 차수로 normalization되었다면 물체의 경계로부터 법선벡터 방향으로 최단거리를 정확하게 계산할 수 있는 거리기반 음함수가 된다. Fig. 4는 무한대와 1차로 normalization된 거리기반 음함수를 예로 하여 normalization condition의 기하학적인 의미를 설명한다. Normalization 차수가 높을수록 물체의 경계로부터 더 먼 위치까지의 거리도 근사하여 계산할 수 있다.

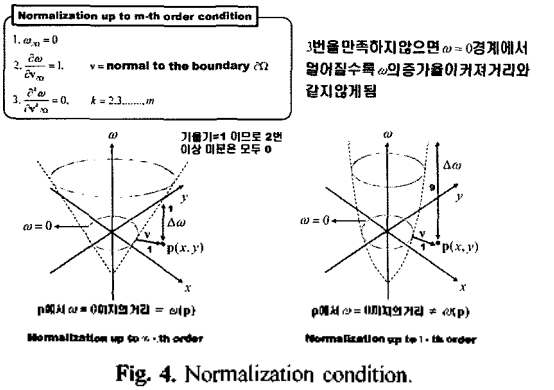

Fig. 4. Normalization condition.

2.1.4 Normalizing method

임의의 음함수가 거리기반 음함수가 되기 위해서는 normalization condition을 만족해야 한다. 그러나 normalization condition을 잘 만족하는 음함수를 찾는 것은 쉬운 일이 아니다. 또한 간단한 기본형상을 표현하는 음함수 뿐만 아니라 임의의 음함수를 거리기반 음함수로 표현해야 하는 경우도 있다. 그러므로 아래와 같이 주어진 임의의 음함수를 거리기반 음함수로 변환할 수 있는 방법이 요구된다[8].

$$f_1 = \frac{f}{\sqrt{f^2 + (\nabla f)^2}}$$

f는 주어진 임의의 음함수이고 f_1는 주어진 음함수 f를 1차로 normalization한 거리기반 음함수이다.

2.2 R-function을 이용한 불리언 연산

2.2.1 정의

R-function은 출력되는 함수값의 부호가 입력된 매개변수들의 부호에 의해서만 결정되는 실수함수이다. 함수값의 부호를 함수의 논리적 특성(logical property)이라고 보면 음수부호는 false, 양수부호는 true라 할 수 있다. R-function은 이러한 논리적 특성(false, true) 사이의 논리연산을 사칙 연산으로 대신 수행할 수 있다. R-function 사이의 다음과 같은 논리연산자와의 관계에 의해 정의된다[3].

$$S_2[f(x,y)] = F[S_2(x), S_2(y)]$$

$$S_2(x) = \begin{cases} 1, & \text{if } x > 0_+ \\ 0, & \text{if } x < 0_- \end{cases}$$

S_2는 함수값을 입력으로 받아서 논리적 특성(1=true, 0=false)을 출력하는 스위칭 함수이다. F는 논리 연산자, 즉 불리언 함수이다. f는 논리 연산자 F에 대응되는 R-function이다. 이때 F는 R-function f의 Boolean companion function이라 한다. 논리 연산자 F는 입력으로 논리적 특성을 받아서 연산하므로 대수적으로 표현하기가 힘들다. 반면 R-function인 f는 입력으로 함수값을 받아서 사칙연산을 수행하므로 대수적으로 쉽게 논리 연산을 할 수 있다. 또한 하나의 논리 연산

자, 즉 Boolean companion function에 대한 R-function은 무수히 많이 존재할 수 있다.

Fig. 5는 불리언 함수 AND와 그에 대응되는 하나의 R-function에 대한 예를 보여준다. 불리언 함수 $F(x, y)$는 논리 연산 AND를 나타내고 이에 대응되는 많은 R-function중에 하나로 $f(x, y)=x+y-\sqrt{x^2+y^2}$을 정의하였다. 임의로 정한 함수 $g(x, y)$는 불리언 함수 AND에 대응되는 R-function이 아닌 것을 알 수 있다.

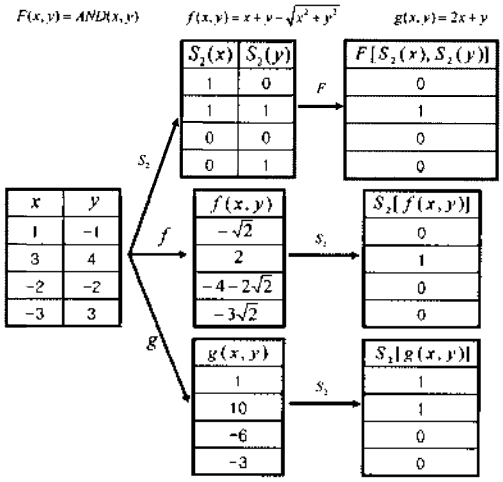

Fig. 5. 불리언 함수 AND와 그에 대응되는 R-function의 예.

2.2.2 음함수 모델의 불리언 연산

음함수로 표현된 기본형상을 R-function을 통해 서로 불리언 연산하여 보다 복잡한 모델을 만들 수 있다. Fig. 6은 R-function을 이용하여 음함수로 표현된 두 개의 원을 불리언 연산하는 예이다. 이 때에 사용된 불리언 연산은 AND와 NOT이고 이를 각각 대응되는 R-function을 이용하여 사칙연산으로 수행한다.

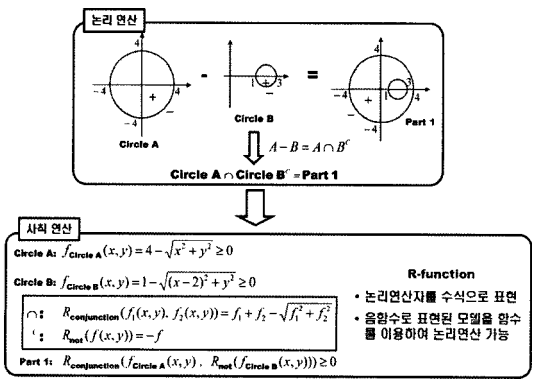

Fig. 6. R-function을 이용한 음함수 모델의 불리언 연산 예.

대표적으로 많이 쓰이는 R-function 시스템은 아래와 같다. + 부호는 OR, − 부호는 AND 불리언 연산을 나타낸다.

$$R_\alpha: \frac{1}{1+\alpha}(x+y \pm \sqrt{x^2+y^2-2\alpha xy})$$

$$R_0^m: (x+y \pm \sqrt{x^2+y^2})(x^2+y^2)^{m/2}$$

$$R_p: x+y \pm (x^p+y^p)^{1/p}$$

2.3 이질속성물체 모델링

2.3.1 이질속성물체의 정의

모델링 과정에서 물체의 기하정보뿐만 아니라 강도, 색상, 재질 등의 물체속성까지 표현해야 하는 경우가 점점 늘고 있다. 물체속성이 일정한 경우는 쉽게 모델링 할 수 있으나 계속 변하는 경우에는 B-Rep 모델링으로는 표현하기 힘들다. 이러한 물체를 이질속성물체, 또는 FGM(Functionally Graded Material)이라 한다.

Fig. 7은 이질속성물체의 예와 모델링 아이디어를 나타낸다. 공구의 앞부분과 손잡이 부분이 접합되는 부분을 FGM으로 정의하였다. FGM 부분의 기하정보는 정면도에서 보는 것과 같이 사다리꼴 형상으로써 쉽게 표현 가능하다. 그러나 물체의 속성은 하나의 속성으로 표현되지 않는다. 우리는 FGM 부분의 물체속성을 표현하기 위해 FGM 내부의 물체속성이 FGM 경계로부터 선형적으로 변한다고 가정한다. 즉 FGM 내부 임의의 위치에서의 물체속성은 이미 물체속성값이 주어진 FGM 경계까지의 최단거리를 통한 선형보간으로 표현될 수 있다[14].

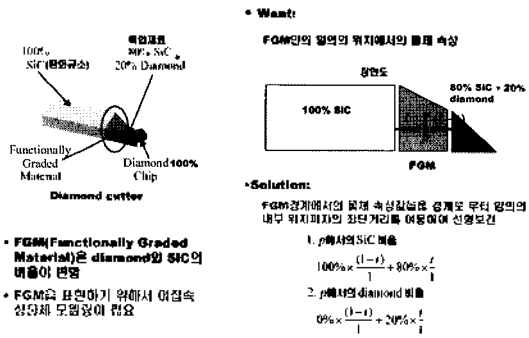

Fig. 7. 이질속성물체의 예와 모델링 아이디어.

2.3.2 Inverse distance weight interpolation

이질속성물체를 표현하기 위한 아이디어로 이미 물체속성이 주어진 경계까지의 최단거리를 통한 선형보

간법을 제시하였다. 즉 거리가 가까운 물체속성의 영향을 많이 받고 거리가 먼 물체속성의 영향을 적게 받는다. 이러한 개념의 선형보간법을 inverse distance weight interpolation 또는 Shepard's method라 하고 다음과 같은 형태를 갖는다[2].

$$f(\mathbf{x}) = \sum_{i=0}^{n} f_i w_i(\mathbf{x})$$

$w_i(\mathbf{x})$는 가중치 함수로서 f_i가 주어진 점 $\mathbf{x}_i$로부터 $\mathbf{x}$까지의 거리에 반비례하는 값을 가진다. $w_i(\mathbf{x})$는 거리에 반비례하는 개념을 이용하여 식 (2)와 같이 정의된다.

$$w_i(\mathbf{x}) = \frac{d_i^{-\mu_i}(\mathbf{x})}{\sum_{j=0}^{n} d_j^{-\mu_j}(\mathbf{x})} \qquad (2)$$

$d_i(\mathbf{x})$는 i번째 주어진 점인 $\mathbf{x}_i$로부터 $\mathbf{x}$까지의 유클리디언(Euclidean) 거리이다. 지수 μ_i는 최종 보간된 함수 $f(\mathbf{x})$의 주어진 점 $\mathbf{x}_i$에서의 미분 성질을 조정한다[5]. $0 \le \mu_i \le 1$ 일 때 최종 보간된 함수는 i번째 주어진 점에서 미분 불가능하다. $\mu_i > 1$일 때 최종 보간된 함수는 i번째 주어진 점에서 $\mu_i - 1$번 미분 가능하다.

2.3.2 Transfinite interpolation

2.3.1에서 보간하고자 하는 함수값이 도메인 상에 서점에 주어져 있을 때의 inverse distance weight interpolation에 대하여 설명하였다. 그러나 Fig. 7에서와 같이 FGM의 경우 보간하고자 하는 함수값이 도메인 상에서 점이 아닌 곡선에 정의 되어 있다. 즉 점에 주어진 함수값을 보간하는 것이 아닌 곡선에 주어진 함수값을 보간할 수 있는 방법이 필요하다. 곡선에 주어진 함수값을 보간하는 방법을 transfinite interpolation이라 한다[11].

Transfinite interpolation 역시 inverse distance weight 개념을 사용한다. 즉 식 (2)을 이용하여 가중치를 계산하게 된다. 그러나 식 (2)는 점 보간, 즉 보간하고자 하는 함수값이 점에 주어져 있을 경우에 사용되는 식이다. 우리가 하고자 하는 transfinite interpolation은 보간하고자 하는 함수값이 곡선에 주어져 있다. 본 연구에서는 모든 물체를 음함수로 모델링 하므로 함수값이 주어진 곡선 역시 음함수로 표현되어 있다. 그러므로 임의의 점에서 가중치를 계산하기 위해서는 점과 음함수 곡선 사이의 최단거리를 계산해야 한다. 그러나 점과 음함수 곡선 사이의 최단거리를 수치적인 방법으로 계산하는 것은 많은 비용이 든다. 거리기반 음함수를 사용하면 이러한 문제를 해결할 수 있다. 점과 곡선사이의 거리를 수치적으로 찾는 것이 아니라 거리기반 음함수에 점을 대입하면 바로 근사된 최단거리를 얻을 수 있기 때문이다. Transfinite interpolation에서 가중치는 다음과 같이 식 (2)를 변형하여 계산된다.

$$w_i(\mathbf{x}) = \frac{\omega_i^{-\mu_i}(\mathbf{x})}{\sum_{j=0}^{n} \omega_j^{-\mu_j}(\mathbf{x})}$$

ω_i는 보간하고자 하는 물체의 속성값이 주어진 i번째 거리기반 음함수 곡선을 나타낸다. Fig. 8은 거리기반 음함수와 transfinite interpolation를 이용하여 이질속성물체를 모델링한 예이다.

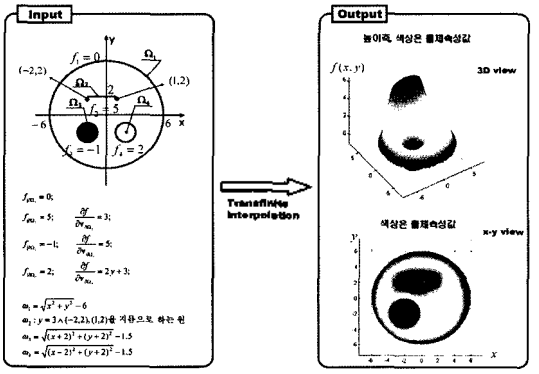

Fig. 8. 이질속성물체 모델링 예.

3. 무요소해석

3.1 개념

무요소법과 유한요소법의 가장 큰 차이점은 해석용 메쉬를 생성하지 않는다는 점이다. 유한요소법은 지금까지 많이 연구되어온 해석 방법으로서 많은 장점들을 가지고 있으나 메쉬를 생성해야 하는 문제를 가지고 있다. 복잡한 형상이나 시간에 따라 물체의 형상이 변하는 경우에 이 메쉬 생성은 전체 해석 과정에 큰 부하를 가져온다. 또한 B-Rep 모델과 해석 모델이 다르기 때문에 모델링과 해석 사이의 피드백을 위해 계속 모델 변환을 해야 하는 문제도 발생한다. 무요소법은 유한요소법의 이러한 메쉬 생성으로부터 파생되는 문제를 해결하는데 초점을 두고 있다.

Table 1에서는 무요소법과 유한요소법의 특징을 비교하여 정리하였다. 기존에 많은 연구가 이루어진 유한요소법에 비하여 무요소법은 많은 연구가 이루어지지 않았다. 현재까지 일반적으로 무요소법이 유한요소법에 비하여 계산속도와 해석의 정확도가 좋지 않

은 것으로 알려져 있다. 반면 무요소법은 메쉬를 생성하지 않음으로써 메쉬 생성으로 인해 파생되는 여러 문제들을 해결할 수 있다는 장점을 가지고 있다. 또한 유한요소법은 메쉬를 생성한 후에 그 메쉬에 따라 형상함수를 정의해야 하지만 무요소법은 메쉬를 생성하지 않으므로 형상함수를 정의할 때에 해석하고자 영역에 영향을 적게 받는다. 즉, 일반적인 삼각함수, 번스타인 기저함수, 비-스프라인 기저함수들을 사용할 수 있다는 장점이 있다. 또한 시간에 따라 움직이는 모델을 해석할 경우 유한요소법은 remeshing해야 하는 문제가 있으나 무요소법은 메쉬를 사용하지 않으므로 보다 유리하다.

Table 1. 무요소법과 유한요소법의 일반적인 특징 비교

	무요소법	유한요소법
메쉬	없다	있다
형상함수	주어진 데이터의 국소해석영역에 영향을 받음	미리 정의된 메쉬에 영향을 받음
계산속도	느리다	빠르다
정확도	부정확하다	정확하다
적응해석 (Adaptive analysis)	쉽다	어렵다

3.2 예제

3.2.1 문제 정의

본 연구에서 사용한 무요소법을 간단한 2차원 열전달 문제를 통해 설명하고자 한다. 이 문제는 Fig. 9와 같이 정의한다. 2개의 거리기반 음함수로 표현된 원을 불리언 하여 가운데가 비어있는 고리형상을 모델링 하였다. 고리의 외부공기의 온도는 1573K로 일정하고 내부의 온도는 1073 K로 일정하다. 즉 고리의 온도와 고리의 내·외부 공기의 온도차에 의해 열이 전달된다. 고리의 열전도도 $\lambda = 7 W/m \cdot K$, 대류 열전달 계수 $\alpha = 100 W/m^2 \cdot K$로 주어져 있다. 우리가 풀고자 하는 미분 방정식, 즉 위의 열전달 현상을 표현하는 물리방정식은 $\nabla^2 T = 0$ 이다. 이 문제는 정상상태(steady-state) 분제로서 시간에 따른 온도변화가 없을 때의 온도분포를 해석한다. 마지막으로 주어진 미분방정식을 풀기 위하여 고리 표면에서 법선 벡터 방향으로의 온도변화율은 고리 내부의 임의의 위치와 표면과의 온도차로 결정된다는 경계 조건이 주어져 있다. 우리는 주어진 모델의 기하형상, 물체의 속성 계수, 경계조건을 이용하여 주어진 미분방정식을 만족하는 온도분포함수 T를 무요소법으로 구하고자 한다.

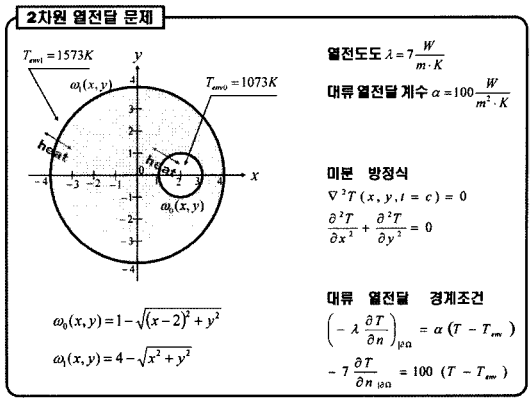

Fig. 9. 2차원 열전달 예제 정의.

3.2.2 Solution structure

Fig. 9에서 정의한 2차원 열전달 문제에서 최종적으로 우리가 구하고자 하는 것은 정상상태의 온도분포함수 T이다. 무요소법과 유한요소법과 마찬가지로 먼저 구하고자 하는 함수의 형태를 가정해야 한다. 유한요소법에서는 trial function으로 구하고자 하는 온도분포함수 T를 가정한다. 이와 같은 개념으로 무요소법에서는 solution structure를 정의하여 구하고자 하는 온도분포함수 T를 가정한다[10,12]. 즉, 온도분포함수 T를 알고 있는 기저함수를 사용하여 표현하는 것이다. 이렇게 구하고자 하는 함수의 형태를 가정하여 정의해야만 주어진 미분 방정식에 맞게 함수를 구할 수 있다.

3.2.2.1 Taylor expansion

Solution structure를 정의할 때에 가장 중요한 개념은 Taylor expansion이다. Taylor expansion은 임의의 위치에서의 함수값과 미분값을 알면 그 함수 전체를 근사하여 정의할 수 있게 한다. Taylor expansion는 식 (3)으로 정의된다.

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \\ + \frac{f^{(a)}a}{n!}(x-a)^n \qquad (3)$$

a에서의 함수값 $f(a)$, 한번 미분값 $f'(a)$, 두번 미분값 $f''(a)$ 등을 알면 임의의 위치 x에서의 함수값을 모두 근사하여 얻을 수 있다. 식 (3)를 f의 a에서의 Taylor expansion이라 한다.

3.2.2.2 일반화된 Taylor expansion

우리가 구하고자 하는 온도분포함수 T는 도메인이

2차원인 함수이다. 그러므로 T에 대한 solution structure를 Taylor expansion을 이용하여 만들 경우 식 (3)을 그대로 사용할 수 없다. 식 (3)은 함수의 도메인이 1차원인 경우이므로 도메인이 다차원인 함수에도 적용 가능한 일반화된 Taylor expansion이 필요하다. 일반화된 Taylor expansion은 거리기반 음함수를 이용하여 얻을 수 있다. 식 (3)을 보면 임의의 위치 x에서의 함수값을 근사할 때 주어진 위치 a까지의 거리가 필요한 것을 알 수 있다. 같은 개념으로 일반화된 Taylor expansion에서는 임의의 위치 x에서 주어진 거리기반 음함수의 경계까지의 거리가 필요하다. 거리기반 음함수는 경계에서 x까지의 거리를 함수값 $\omega(x)$로 출력하므로 쉽게 거리를 계산할 수 있다. 식 (4)는 거리기반 음함수로 표현된 도메인의 경계에서의 함수값과 m번 미분값까지 주어졌을 때의 일반화된 Taylor expansion이다[11].

$$f(\mathbf{x}) = f_0^* + \sum_{k=1}^{m} \frac{1}{k!} f_k^* \omega(\mathbf{x})^k + O(\omega^{m+1}) \tag{4}$$

아래첨자는 미분의 회수를 나타낸다. f_0는 $\omega = 0$에서의 함수값, f_1은 $\omega = 0$에서의 법선벡터 방향으로의 한번 미분값이다. 윗첨자 f^*는 normalizer를 나타낸다. f^*는 f의 ω에 대한 normalizer이다. Taylor expansion에서 계수들은 상수이어야 하므로 normalizer를 이용하여 일반화된 Taylor expansion의 계수를 $\omega = 0$에서 법선벡터 방향으로 상수화할 수 있다. $f(\mathbf{x})$의 ω에 대한 normalizer $f^*(\mathbf{x})$는 다음과 같이 정의된다[11].

$$\begin{aligned} f^*(\mathbf{x}) &= f(\mathbf{x} - \omega(\mathbf{x})\nabla\omega(\mathbf{x})) \\ &= f(\mathbf{x}) - \omega(\mathbf{x})\nabla\omega(\mathbf{x}) \cdot \nabla f(\mathbf{x}) \end{aligned} \tag{5}$$

3.2.2.3 Constructing solution structure

Solution structure는 거리기반 음함수로 주어진 기하정보와 경계조건을 이용하여 일반화된 Taylor expansion으로 만들어 진다. 이 때 주어진 경계조건을 만족하도록 가정해야 한다.

$$\left(\frac{\partial T}{\partial n}\right)_{|\partial\Omega} = \frac{100\,T_{env}}{7} - \frac{100}{7}T \tag{6}$$

이 경계조건은 도메인의 경계에서의 법선벡터 방향으로의 미분값을 나타낸다. 그러므로 우리는 이 경계조건을 만족하는 solution structure를 만들기 위해 도메인의 경계에서의 일반화된 Taylor expansion를 이용하여 다음과 같이 정의한다[10,12].

$$T = T_0^* + T_1\omega + \omega^2\Phi_2 \tag{7}$$

T_0는 도메인의 경계에서의 함수값, T_1은 도메인의 경계에서의 법선벡터 방향으로의 한번 미분값이다. 이 때 주어진 경계조건인 식 (6)가 바로 T_1이 됨을 알 수 있다.

Solution structure를 정의할 때에 모르는 항은 기저함수의 선형조합으로 가정한다. 경계조건에 의해 도메인의 경계에서의 한번 미분값까지만 주어져 있으므로 두번 이상의 미분값들은 식 (4)에서 에러항으로 남게 된다. 보통의 경우 이 에러항은 표현하지 않으나 우리는 해석의 정확도를 높이고자 식 (7)의 에러항을 기저함수의 선형조합 Φ_2로 근사한다.

$$\Phi_2 = \sum_{i=0}^{n} c_{2,i}\chi_i$$

χ_i는 기저함수로서 삼각함수, 번스타인 함수, 비-스프라인 함수 등을 모두 사용할 수 있으며 본 연구에서는 비-스프라인 함수를 사용한다[6,9]. $c_{2,i}$는 기저함수의 계수이다. 기저함수는 이미 알고 있으므로 이 계수들이 결국 solution structure에서 구해야 하는 미지수가 된다. T_1은 경계조건으로 주어져 있으나 T_0는 문제에서 주어지지 않았으므로 에러항과 마찬가지로 기저함수의 선형조합으로 가정하여 표현한다. T_0는 Φ_1으로 다음과 같이 가정한다.

$$\Phi_1 = \sum_{i=0}^{n} c_{1,i}\chi_i$$

식 (7)를 살펴보면 일반화된 Taylor expansion을 위해 T_0를 ω에 대하여 normalize하여 T_0^*로 변형하였다. T_0^*는 식 (5)에 의해 다음과 같이 정의된다.

$$T_0^* = T_0 - \omega\nabla\omega \cdot \nabla T_0$$

앞서 우리는 T_0를 Φ_1으로 가정하였으므로 T_0^*는 다음과 같이 다시 정리된다.

$$T_0^* = \Phi_1 - \omega\nabla\omega \cdot \nabla\Phi_1$$

최종적인 solution structure의 형태는 다음과 같다.

$$\begin{aligned} T = {} & \Phi_1 - \omega\nabla\omega \cdot \nabla\Phi_1 + \\ & \left(\frac{100\,T_{env}}{7} - \frac{100}{7}\Phi_1\right) + \omega^2\Phi_2 \end{aligned} \tag{8}$$

원래의 경계조건은 식 (6)으로 주어져 있으나 경계

조건에서의 T는 도메인의 경계에서의 온도를 나타내므로 앞서 정의한 Φ_1으로 치환하였다.

Solution structure 식 (8)을 살펴보면 물체의 주위 공기 온도를 나타내는 $T_{em}(x, y)$항이 존재한다. 이는 임의의 위치 (x, y)에서의 물체의 주위 공기 온도를 나타내는 함수이다. 그러나 물체의 주위 공기 온도는 고리의 안쪽에서 1073 K, 바깥쪽에서 1573 K로만 주어져 있고 고리 내부 임의의 위치에서는 정의 되지 않는다. 그러므로 고리 내부의 임의의 위치에서도 주위 공기 온도의 영향을 나타낼 수 있는 $T_{env}(x, y)$를 정의하여야 한다. Transfinite interpolation을 이용하여 $T_{env}(x, y)$를 다음과 같이 정의할 수 있다.

$$W_0(x,y) = \frac{\frac{1}{\omega_0(x,y)}}{\frac{1}{\omega_0(x,y)}+\frac{1}{\omega_1(x,y)}},$$

$$W_1(x,y) = \frac{\frac{1}{\omega_1(x,y)}}{\frac{1}{\omega_0(x,y)}+\frac{1}{\omega_1(x,y)}} \tag{9}$$

$$\begin{aligned} T_{env}(x,y) &= W_0(x,y)\times T_{env0} + W_1\times(x,y)T_{env1} \\ &= W_0(x,y)\times 1073 + W_1(x,y)\times 1573 \end{aligned}$$

$\omega_0 = 0$, $\omega_1 = 0$은 각각 T_{env0}, T_{env1}가 주어진 도메인의 내·외부의 경계를 나타낸다.

3.2.3 Ritz method

우리는 구하고자 하는 온도분포함수 T를 일반화된 Taylor expansion과 경계조건을 이용하여 식 (8)과 같이 solution structure로 정의하였다. 이 때 T는 열전달 현상을 물리적으로 표현하는 미분방정식 $\nabla^2 T = 0$을 만족해야 한다. 즉, 주어진 미분방정식을 만족하는 solution structure 기저함수의 계수들을 구해야 한다.

미분방정식을 푸는 방법은 Galerkin method, Ritz method, Least square method 등의 여러 방법이 있다. 본 연구에서는 Ritz method를 이용하여 주어진 미분방정식을 풀고자 한다. Ritz method는 주어진 미분방정식을 직접 풀지 않고 이에 대응되는 potential energy functional 최소화를 이용하는 방법이다. 우리가 예제에서 풀고자 하는 미분 방정식은 다음과 같다.

$$\nabla^2 T(x,y) = 0$$

이는 라플라스(Laplace) 방정식이므로 다시 정리하면 다음과 같다.

$$\frac{\partial^2 T(x,y)}{\partial x^2} + \frac{\partial^2 T(x,y)}{\partial y^2} = 0 \tag{10}$$

식 (10)는 우리가 원하는 온도분포함수 T에 대하여 x, y 방향으로 각각 두번 편미분하는 항으로 이루어져 있다. 즉 직관적으로 생각하였을 때 T를 얻기 위해서는 식 (10)를 두번 적분하는 개념이 필요하다.

Ritz method는 식 (10)에 대한 potential energy functional를 이용한다. functional은 입력으로 function을 받아서 실수를 출력하는 것을 말한다. 식 (10)에 대한 Potential energy functional $I[T(x,y)]$은 다음과 같다[11].

$$I[T(x,y)] = \iint_{\Omega}\left(\frac{\partial T}{\partial x}\right)^2 + \left(\frac{\partial T}{\partial y}\right)^2 dxdy \tag{11}$$

Ritz method는 미분방정식을 직접 풀기 않고 그에 대응되는 potential energy functional 최소화문제로 미분방정식을 푼다. 즉, 식 (10)를 직접 푸는 것과 식 (11)을 최소화하는 문제를 푸는 것은 결과적으로 서로 동치이다. 식 (10)을 살펴보면 적분기호 안에 T에 대한 한번 편미분 항이 있는 것을 알 수 있다. 즉 개념적으로 식 (10)는 두번 미분된 식을 두번 적분해야 하는 반면 식 (11)은 한번 미분된 식을 한번 미분하는 형태이므로 더 쉽게 풀 수 있다.

Ritz method를 이용하여 미분방정식을 풀기 전에 solution structure 식 (8)을 미지수 $c_{1,i}, c_{2,i}$가 있는 항과 없는 항으로 다음과 같이 분리한다.

$$T_0 = \sum_{i=0}^{n} c_{1,i}\left(\chi_i - \omega\nabla\omega\cdot\nabla\chi_i - \frac{100}{7}\omega\chi_i\right) + \omega^2\sum_{i=0}^{n} c_{2,i}\chi_i$$

$$T_1 = \frac{100 T_{env}}{7}\omega$$

주어진 미분 방정식 $\nabla^2 T = 0$에 대한 potential energy functional은 다음과 같다.

$$I = -\iint_{\Omega}\lambda(\nabla T_0)^2 d\Omega - 2\iint_{\Omega}\lambda(\nabla T_0)(\nabla T_1)d\Omega - \int_{\partial\Omega}\alpha T_0^2 dS + 2\int_{\partial\Omega}\alpha T_0 T_{env} dS \tag{12}$$

우리는 이제 위 식을 최소화 하는 문제를 풀어야 한다. 일반적으로 최소화 문제는 극값을 찾는 문제이다. 즉, 최소화 하고자 하는 함수를 함수의 미지수로 미분하여 0이 되는 점을 찾으면 된다. 같은 개념으로 식 (12)을 미지수 $c_{1,i}, c_{2,i}$로 편미분하여 0이 되도록

한다. 예를 들어 미지수 $c_{1,1}$으로 편미분하여 0의 되도록 하는 식은 다음과 같다.

$$\frac{\partial I}{\partial c_{1,1}} = -2\lambda \iint_{\Omega} (\nabla T_0)\frac{\partial(\nabla T_0)}{\partial c_{1,1}} - (\nabla T_1)\frac{\partial(\nabla T_0)}{\partial c_{1,1}} d\Omega$$
$$-2\alpha \int_{\partial\Omega} T_0 \frac{\partial(\nabla T_0)}{\partial c_{1,1}} - T_{env}\frac{\partial(\nabla T_0)}{\partial c_{1,1}} dS = 0$$

위와 같이 모든 미지수에 대하여 편미분하여 0이 되도록 하면 최종적으로 미지수에 대한 linear system을 만들 수 있다. 이 linear system을 풀면 미지수 $c_{1,i}, c_{2,i}$를 모두 찾을 수 있다[9].

3.2.4 결과

Fig. 10은 예제에서 정의한 고리형상의 2차원 열전달 문제를 해석한 결과이다. 기저함수로는 bicubic 비-스프라인 함수를 사용하였고 조종점은 x, y 방향으로 10개씩 사용하였다. 즉 식 (8)로 정의된 solution structure에서 미지수인 $c_{1,i}, c_{2,i}$가 각각 100개가 된다.

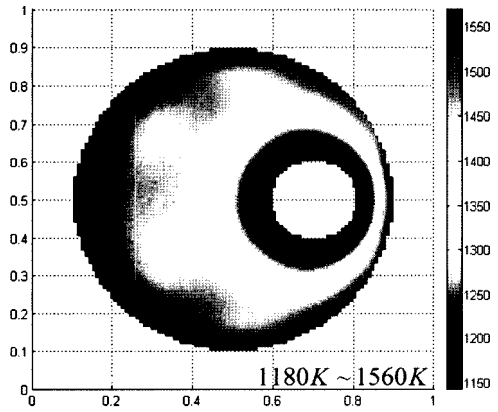

Fig. 10. 2차원 열전달 예제 해석 결과.

3.2.5 GPU 프로그래밍을 이용한 무요소법

무요소법은 유한요소법에 비하여 해석속도가 느린 것으로 알려져 있다. 유한요소법은 각각의 메쉬마다 단순한 선형의 기저함수가 사용되는 반면 무요소법의 기저함수는 번스타인, 비-스프라인 함수 등을 사용하므로 미분과 적분 등의 계산이 보다 어렵다. 본 연구에서는 GPU 프로그래밍을 이용하여 무요소법의 계산속도를 향상시키고자 한다.

3.2.5.1 GPU 프로그래밍 개념

GPU는 Graphics Processing Unit으로서 그래픽카드 안에서 그래픽 명령어 처리를 담당한다. CPU (Central Processing Unit)와 대비하여 설명할 수 있는데 CPU는 컴퓨터에서 모든 명령어를 처리하도록 범용으로 설계되었고 GPU는 그래픽 명령어만을 빠르게 처리할 수 있도록 설계되었다. Table 2는 CPU 프로그래밍과 GPU 프로그래밍을 비교한 표이다. CPU를 이용하는 프로그램을 C, C++, JAVA 등의 고수준 언어로 작성하듯이 GPU를 이용하는 프로그램은 CG, GLSL, HLSL등을 이용한다. 이 때 GPU에서 실행되는 프로그램을 shader라 하며 vertex shader와pixel shader가 있다.

Table 2. CPU프로그래밍과 GPU프로그래밍의 비교

	CPU 프로그래밍	GPU 프로그래밍
실행속도	느림	빠름
용도	범용	그래픽 전용
저수준언어	어셈블리	shader 어셈블리
고수준언어	C, C++, JAVA,	CG, GLSL, HLSL

Table 3은 고수준 shader 언어의 종류와 특징을 정리한 것이다. 본 연구에서는 HLSL을 이용하였다.

Table 3. 고수준 shader 언어의 종류와 특징

	CG (C for Graphics)	GLSL(OpenGL Shader Language)	HLSL(High Level Shader Language)
특징	· NVIDIA 고유의 형식 · NVIDIA의 그래픽카드만 사용 가능	· OpenGL 환경 · .NET, UNIX 기반 컴파일 지원	· DirectX9 환경 · VC++ 6.0 컴파일러 지원

3.2.5.2 프로그래밍 가능한 pixel shader

Pixel shader는 원래 그래픽용으로는 텍스쳐 맵핑을 수행하는데 사용된다. 이러한 pixel shader는 OpenGL이나 DirectX에서 그래픽 라이브러리 함수로 제공된다. 그러나 vertex shader와 pixel shader 모두 shader 언어를 이용하면 프로그래머가 원하는 프로그램으로 작성이 가능하다. 본 연구에서는 프로그래밍 가능한 pixel shader를 이용하여 GPU를 그래픽 처리가 아닌 프로그래머가 원하는 임의의 함수계산에 이용하고자 한다. 특히 pixel shader를 이용하면 CPU 프로그래밍에서는 for문이 두 번 필요한 연산을 1프레임 가시화로 for loop없이 빠르게 연산 가능하다. Fig. 11은 프로그래밍 가능한 pixel shader를 임의의 함수

계산에 이용하는 개념을 일반적인 텍스쳐 맵핑 과정과 대비하여 설명한다.

Pixel shader를 임의의 함수 계산에 사용할 때에도 텍스쳐 맵핑에 사용될 때와 같은 개념을 사용한다. 2차원 텍스쳐 이미지는 내부적으로는 텍스쳐 맵핑될 RGB 값이 저장되어 있는 2차원 배열이다. 즉, 텍스쳐에 함수계산에 사용될 입력값들을 2차원 배열로 저장할 수 있다. 화면의 픽셀도 역시 내부적으로는 화면에 출력될 RGB 값이 저장되는 메모리상의 2차원 배열이다. 즉, 화면의 픽셀이 저장되는 2차원 배열은 계산된 함수값을 저장하는데 쓰인다. Pixel shader에는 우리가 계산하고자 하는 함수를 shader 언어로 작성한 프로그램이 된다. 가시화하려는 기하정보는 가시화 하였을 때 화면에 맞는 크기가 되는 사각형으로 정의한다. 그리고 입력이 되는 텍스쳐의 해상도와 출력이 되는 화면의 해상도를 갖게 하면 텍셀과 픽셀이 서로 1:1로 대응되어 입력값이 저장된 한 텍셀이 pixel shader를 통해 함수값이 계산되면 동일한 위치의 픽셀에 저장되게 된다. 즉 첫번째 텍셀의 함수값은 첫번째 픽셀에 저장되고 두번째 텍셀의 함수값은 두번째 픽셀에 저장된다. Fig. 11에서는 13 by 10의 해상도를 사용하였으므로 총 130개의 입력값을 13 by 10의 for loop를 사용하지 않고 1프레임에 처리할 수 있다.

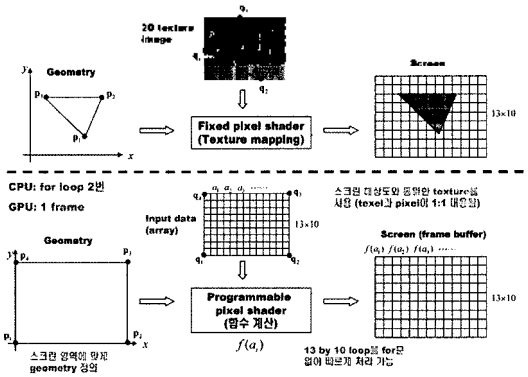

Fig. 11. 프로그래밍 가능한 pixel shader 개념.

Table 4는 실제로 2097152개의 점을 입력으로 하여 아래와 같이 2개의 음함수를 불리언 하는 계산을 수행하였을 때 GPU와 CPU의 계산속도를 비교하였다. CPU는 인텔 펜티엄4 2.6GHz, GPU는 Geforce 6600GT를 사용하였다. GPU를 사용했을 때가 약 7.306452배 정도 더 빠른 계산결과를 얻을 수 있었다. Fig. 12는 실험에 사용된 pixel shader 코드이다.

$$\omega(x,y) = \omega_0(x,y) \wedge \overline{\omega_1}(x,y)$$

$$\omega_0(x,y) = \frac{0.4^2 - (x-0.5)^2 - (y-0.5)^2}{2\times 0.4}$$

$$\omega_1(x,y) = \frac{0.1^2 - (x-0.7)^2 - (y-0.5)^2}{2\times 0.1}$$

Table 4. GPU와 CPU의 계산시간 비교

계산방법	GPU (pixel shader)	CPU
시간(s)	0.062	0.453

```
sampler tex;

struct PS_INPUT
{
    float2 texcoord : TEXCOORD0;
};

struct PS_OUTPUT
{
    vector value : COLOR0;
};

// compute circle, distance function
float circle(in float xc, in float yc, in float r, in float x, in float y)
{
    return (r*r - (x - xc)*(x - xc) - (y - yc)*(y - yc)) / (2*r);
}

// R-function, AND
float conjunction(in float f1, in float f2)
{
    return f1 + f2 - sqrt(f1*f1 + f2*f2);
}

// R-function, NOT
float not(in float f)
{
    return -f;
}

float p1_x, p1_y, p2_x, p2_y;          // input coordinates
float x1 = 0.5, y1 = 0.5, r1 = 0.4;    // circle1 (center, radius)
float x2 = 0.7, y2 = 0.5, r2 = 0.1;    // circle2 (center, radius)

// compute omega(p1), omega(p2)
PS_OUTPUT gpu_main(PS_INPUT input)
{
    PS_OUTPUT output = (PS_OUTPUT)0;

    // input coordinates
    vector tmp = tex2D(tex, input.texcoord);
    p1_x = tmp[0];
    p1_y = tmp[1];
    p2_x = tmp[2];
    p2_y = tmp[3];

    // compute omega(p1), omega(p2)
    float w0_1 = circle(x1, y1, r1, p1_x, p1_y);
    float w1_1 = circle(x2, y2, r2, p1_x, p1_y);
    float w_1 = conjunction(w0_1, not(w1_1));

    float w0_2 = circle(x1, y1, r1, p2_x, p2_y);
    float w1_2 = circle(x2, y2, r2, p2_x, p2_y);
    float w_2 = conjunction(w0_2, not(w1_2));

    // return
    output.value[0] = w_1;
    output.value[1] = w_2;

    return output;
}
```

Fig. 12. Pixel shader 코드 예.

4. 실제 적용 예: 파이프 디자인

4.1 문제 정의 및 가설

Fig. 13과 같이 난방용 배관을 설치하고자 한다. 본 연구에서 제안하는 통합된 모델링과 해석 프레임워크를 통해 동일한 파이프 재질과 동일한 온도의 온수를 공급하면서 파이프의 열효율(파이프 외부표면의 온도)을 높이고자 한다. 80°C의 온수가 파이프 내부로 일정하게 공급되고 파이프 외부의 온도는 항상 15°C이다. 파이프의 열전도도는 400 W/m·K이고 외부표면

과 내부표면의 열전달 계수는 각각 30 W/m²·K와 150 W/m²·K이다.

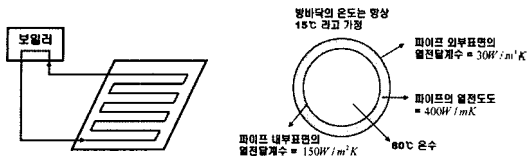

Fig. 13. 난방용 배관 설치 문제 정의.

온수온도와 파이프의 재질을 변화시키지 않고 파이프의 기하형상만 변화시켜서 파이프의 외부표면온도를 높이고자 한다. 파이프의 내부에서 온수와 닿는 면적을 넓게 하면 파이프 외부표면온도가 높아질 것이라는 가설을 세우고 이를 검증하고자 한다.

4.2 가설 검증

파이프의 내부에서 온수와 닿는 면적을 넓게 하기 위하여 파이프의 내부에 핀을 설치한다. Fig. 14와 같이 3가지 형상으로 파이프를 모델링하였다.

3가지 경우의 파이프 모델에 대하여 각각 무요소법

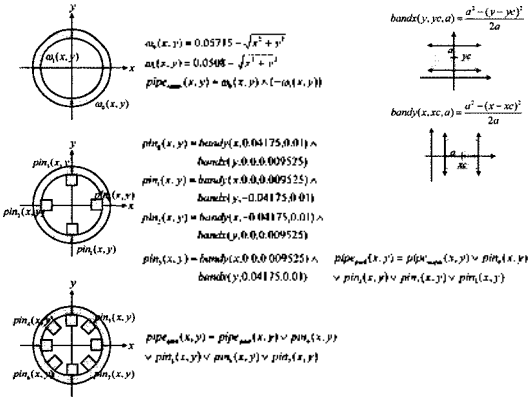

Fig. 14. 파이프의 형상 모델링.

으로 해석하여 파이프 외부표면의 온도를 계산하고자 한다. 이 때 열전달을 나타내는 미분 방정식은 $\nabla^2 T = 0$ 이다. Solution structure는 $T = T_0 + T_1$ 의 형태로 다음과 같이 정의된다. T_0는 미지수가 있는 부분이고 T_1은 없는 부분이다.

$$T_0 = \sum_{i=0}^{n} c_{1,i}\left(\chi_i - \omega\nabla\omega\cdot\nabla\chi_i - \frac{\alpha}{400}\omega\chi_i\right) + \omega^2\sum_{i=0}^{n} c_{2,i}\chi_i$$

$$T_1 = \frac{\alpha T_{env}}{400}\omega$$

물체 외부의 온도 $T_{env}(x, y)$와 물체의 열전달 계수 $\alpha(x, y)$는 식 (9)과 같이 transfinite interpolation을 통하여 만들어 진다.

Fig. 15는 3가지 경우의 파이프 모델에 대하여 해석한 결과이다. 파이프의 내부에 핀을 설치하면 파이프 외부표면의 온도가 더 높아지는 것을 알 수 있다. 즉, 온수온도와 파이프 재질을 바꾸지 않고 파이프 내에 핀을 설치하면 파이프 외부표면의 온도를 더 높아져서 열효율이 상승한다.

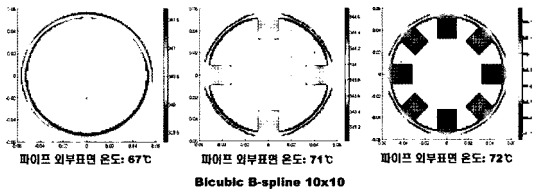

Fig. 15. 파이프 모델 해석 결과.

5. 결론 및 향후 연구

기존의 B-Rep 모델링과 유한요소법 프레임워크는 모델링과 해석 사이에서 메쉬를 생성해야 한다. 따라서 복잡하거나 시간에 따라 움직이는 모델을 해석할 경우 메쉬 생성에 많은 비용이 든다. 해석결과를 모델의 형상에 다시 피드백으로 반영할 때에도 해석용 메쉬를 B-Rep 모델링에서 사용할 수 있는 곡면정보로 다시 변환해야 한다. 그리고 B-Rep 모델링으로는 이질속성물체를 표현하기 힘들다.

R-function을 이용한 거리기반 음함수 모델링과 무요소법은 위와 같은 기존의 문제를 보완할 수 있는 한 방법이다. 거리기반 음함수 모델은 transfinite interpolation을 이용하여 쉽게 이질속성물체를 표현할 수 있다. 또한 무요소법은 거리기반 음함수 모델로부터 메쉬를 생성하지 않고 바로 해석할 수 있으므로 유한요소법이 갖는 메쉬 생성, remeshing 등과 같은 모델 변환 문제가 없다. 즉, 거리기반 음함수 모델링과 무요소법은 통합된 모델링과 해석 프레임워크를 제공한. 그러나 무요소법은 메쉬로 인한 문제가 없는 대신 유한요소법 비하여 해석과정에서 많은 계산이 필요하다. 본 연구에서는 이를 해결하기 위해 GPU 프로그래밍을 무요소법에 적용하였다. 해석과정에서 필요한 함수를 pixel shader로 프로그래밍한 후 GPU로 계산하여 CPU보다 빠른 계산을 수행하였다.

향후 본 연구의 통합된 모델링과 해석 프레임워크에대하여 다음과 같은 문제들을 해결해야 한다. R-function을 이용한 불리언 반으로는 freeform object를

음함수로 표현하는데 한계가 있으므로 이를 해결할 수 있는 연구가 필요하다. 또한 시간에 따라 움직이는 모델에 대한 무요소법도 연구되어야 할 것이다.

감사의 글

본 연구는 한국과학재단 특정기초연구 R01-2005-000-11257-0 지원에 의하여 수행되었습니다. 또한 Dr. Igor Tsukanov께서 본 연구에 대하여 함께 토의하고 제안을 해준 점에 대해 감사 드립니다.

참고문헌

1. Rvachev, V. L., *Geometric Applications of Logic Algebra*, Naukova Dumka, 1967, In Russian.
2. Shepard, D., "A Two-dimensional Interpolation Function for Irregularly Spaced Data", In *Proceedings 23rd ACM National Conference*, pp. 517-524, 1968.
3. Shapiro, V., Theory of R-functions and Applications: A Primer, Tech. Report TR91-1219, Computer Science Department, Cornell University, Ithaca, NY, 1991.
4. Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P., *Numerical Recipes in C*, Cambridge Unversity Press, second edition, 1992.
5. Hoschek, J. and Lasser, D., *Fundamentals of Computer Aided Geometric Design*, A K Perters, 1993
6. Farin, G., *Curves and Surfaces for Computer Aided Geometric Design*, 4th edition, Academic Press, 1996.
7. Bloommenthal, J., *Introduction to Implicit surfaces*. Morgan Kaufmann Publishers, 1997.
8. Shapiro, V. and Tsukanov, I., Implicit Functions with Guaranteed Differential Properties, *Proceedings of the Fifth ACM Symposium on Solid Modeling and Applications*, Ann Arbor, MI, June 1999.
9. Farin, G. and Hansford, D., *The Essentials of CAGD*, AK Peters, 2000.
10. Rvachev, V. L., Sheiko, T. I., Shapiro, V. and Tsukanov, I., "On Completeness of RFM Solution Structures", *Computational Mechanics* 25, pp. 305-316, 2000.
11. Rvachev, V. L., Sheiko, T. I., Shapiro, V. and Tsukanov, I., "Transfinite Interpolation Over Implicitly Defined Sets", *Computer-Aided Geometric Design*, Vol. 18, No. 4, pp. 195-220, 2001.
12. Tsukanov and Shapiro, V., "The Architecture of SAGE - A Meshfree System Based on RFM", *Engineering with Computers*, Vol. 18, No. 4, pp. 295-311, 2002.
13. Biswas, A. and Shapiro, V., "Approximate Distance Fields with Non-Vanishing Gradients", *Graphical Models*, Vol. 66, Issue 3, May, pp 133-159, 2004.
14. Biswas, A., Shapiro, V. and Tsukanov, I., "Heterogeneous Material Modeling with Distance Fields", *Computer-Aided Geometric Design*, Vol. 21, pp. 215-242, 2004.
15. Dokken, T., Hagen, T. R. and Hjelmervik, J. M., "The GPU as a High Performance Computational Resource", *Proceeding from Spring Conference on Computer Graphics 2005*, Budmerice Castle, Bratislava, Slovakia, May 12-14 2005.
16. General-Purpose Computation Using Graphics Hardware. In *http://www.gpgpu.org/*.
17. 신기훈, 이진구, "임의 형상의 복합재 모델링을 위한 CSG 기반 표현", 한국 CAD/CAM 학회 논문집, 제11권, 제4호, pp. 235-234, 2006.
18. 신기훈, 김주한, "복합재 형상의 FEA기반 설계를 위한 통합 CAD 시스템", 한국 CAD/CAM 학회 논문집, 제10권, 제5호, pp. 328-338, 2005.
19. 유병현, 한순흥, "선택저장 자료구조를 이용한 복합 다양체 모델의 불리안 작업", 한국 CAD/CAM 학회 논문집, 제5권, 제4호, pp. 293-300, 2000.

신 헌 주

2004년~2006년 서울대학교 계산과학과 협동과정 석사
2006년~현재 아이너스기술, 소프트웨어엔지니어
관심분야: 음함수 모델링 및 해석, 컴퓨터 그래픽스

김 태 완

1985년 한양대학교 산업공학사
1993년 미국 Arizona State Univ., Computer Science 석사
1996년 미국 Arizona State Univ., Computer Science 박사
1996년~1999년 미국 SDRC소프트웨어엔지니어
1999년~2001년 서울대학교 정밀기계설계공동연구소 특별연구원
2001년~2002년 세종대학교 컴퓨터부 디지털콘텐츠학과 조교수
2003년~현재 서울대학교 공과대학 조선해양공학과 부교수
관심분야: NURBS 곡선과 곡면, CAGD, 컴퓨터그래픽스, 디지털 콘텐츠제작

신 동 우

1981년 서울대학교 자연과학대학 수학과 이학사
1983년 서울대학교 대학원 수학과 이학 석사
1984년~1985년 삼성전자 사원
1991년 Purdue Univ., 응용수학 박사
1991년~1992년 Italy Institute of Numerical Analysis, Visiting Professor
1993년~현재 서울대학교 자연과학대학 수학과 교수
1999년~2000년 호주 University New South Wales International Fellow
2003년~2005년 서울대학교 협동과정 계산과학전공 주임
2006년~2007년 미국 Purdue University, Visiting Professor
관심분야: 응용수학 Applied Mathematics (수치해석, 역문제, 계산과학 Numerical analysis, Inverse problems, computational science)