論文 J. of The Korean Society for Aeronautical and Space Sciences 43(12), 1079-1088(2015)
DOI:http://dx.doi.org/10.5139/JKSAS.2015.43.12.1079
ISSN 1225-1348(print), 2287-6871(online)

# 인공위성 열해석을 위한 복사형상계수 계산기법의 병렬화 및 성능향상 기법 연구

김민기*

# Studies of Parallelism and Performance Enhancements of Computing View Factor for Satellite Thermal Analysis

Min-Ki Kim*
Korea Aerospace Research Institute

## ABSTRACT

Parallelism and performance enhancement of calculating view factors in KSDS developed by KARI is introduced in this paper. View factor is an essential parameters of radiation thermal analysis for a spacecraft, and the amount of computation of them is not negligible. Especially, independent integration of view factors at each position of the orbit because the relative displace between solar panel and main body of a satellite varies with the position on the orbit. This paper introduces a range of parallelism of computing view factor and their performance, detection of obstructions by spatial search algorithm based on KD-Tree, and the reduction of the calculation of view factors of a satellite with relative motion between solar panel and main body, called updating fractional view factor matrix, for satellite thermal analysis.

## 초 록

본 연구는 한국항공우주연구원에서 개발한 위성종합설계 SW 내의 복사계수 계산 프로그램의 병렬화 및 성능향상에 대해 논의한다. 복사계수는 복사열전달이 포함된 인공위성의 열해석을 수행하기 위한 필수적인 전초 단계로서 그 자체적인 계산량 또한 상당하다. 특히 위성 궤도상 시간에 따라 태양전지판과 본체의 상대변위가 변하기에 시간 별 독립적인 복사계수의 계산이 필요하다. 본 논문은 복사형상계수 병렬화 방법과 그 성능, KD-Tree 기반 차폐 탐색 알고리즘 및 태양전지판과 본체의 상대변위 변화에 따른 부분 복사형상계수 행렬 갱신이라고 지칭하는 계산량 저감 기법에 대해 논한다.

**Key Words :** KSDS(위성종합설계 소프트웨어), View Factor(복사형상계수), OpenMP(오픈MP), KD-Tree(KD트리), Updating Partial View Factor Matrix(부분 복사형상계수 행렬 갱신)

† Received : August 12, 2015 Revised : October 13, 2015 Accepted : October 30, 2015
* Corresponding author, E-mail : mkkim12@kari.re.kr

## Ⅰ. 서 론

극한의 온도변화에 놓이게 되는 인공위성의 설계에 있어서 복사열전달이 포함된 열해석은 무척 중요한 부분이다. 이에 항공우주연구원은 복사열전달을 포함한 위성설계에 필요한 주요 해석기들을 단일 사용자 인터페이스에 통합한 인공위성 종합설계 소프트웨어인 KSDS(KARI Satellite Design Software)[1-10]를 개발하고 있다. 해당 S/W는 복사열전달을 위한 복사형상계수(View Factor), 복사열교환계수(Radiative Thermal Exchange Factor), 태양열류벡터(Solar Heat Flux)를 계산하는 해석기 및 이를 입력받아 시간에 따른 온도변화를 해석하는 열해석 솔버를 내장하고 있다.

위성종합설계 소프트웨어 KSDS는 인공위성의 설계에 필수적인 각종 해석 모듈들[2-9] 및 이들을 쉽게 사용할 수 있는 전후처리 기능을 갖춘 통합된 사용자 인터페이스를 제공하는, 인공위성을 포함한 우주구조물 설계 및 해석에 활용되는 프로그램이다. KSDS에는 인공위성의 설계를 위한 간단한 형상 모델링[1]을 포함하여 궤도 결정[2], 열해석[3] 및 이를 위한 복사형상계수와 태양열류량 계산모듈[4], 오염해석(Outgassing)[5], 외란토크 계산 모듈[6], 우주방사선 조사량 예측 및 태양전지판 성능예측 모듈[7], 통신링크 버짓 계산 모듈[8], 전력해석 모듈[9] 등으로 구성되어 있으며 이들 해석모듈들의 사용자 인터페이스 및 전후처리 모델링 기능을 갖추고 있다. 최근에 구조해석 솔버[10-13], 모델러 및 열-구조 연계해석을 위한 온도변환 코드[14]를 개발하였다. Fig. 1은 위성종합설계 소프트웨어의 전반적인 구성을 나타내고 있다.

복사열전달의 해석을 위해서는 복사형상계수 및 복사열교환 계수의 계산이 필수적이다. 복사형상계수의 계산 방법에는 크게 직접 수치적분법과 몬테카를로 광선추적법(Monte-Carlo Ray Tracing) 방법이 있는데, KSDS에서는 직접 수치적분법을 적용하였다[4,15,16]. 사용된 방법은 복사계수 계산의 대상이 되는 두 요소 간 입체각을 기준으로 격자를 세분화하여 수치적분을 수행하는 것이다[17-22]. 직접 수치적분법은 몬테카를로 광선추적법에 비해 복사형상계수의 상호성(Reciprocity)이 보존되고 비교적 그 결과가 정확하다는 데 그 장점이 있다. 하지만 복사계수의 수치적분을 위해서 전체 요소수의 제곱에 비례하는 계산량이 요구되기에 이를 극복하기 위해 구현 시 많은 고려를 해야 한다.

본 논문은 KSDS에 구현된 복사형상계수 계산 프로그램의 성능 향상을 위해 세 가지 기법을 도입하였다. 첫 번째는 OPENMP[23] 병렬화[24], 두 번째는 KD-Tree 기반 공간 탐색 차폐 탐색 알고리즘[25-27], 세 번째는 태양전지판과 본체의 상대적 움직임이 있을 때 적용할 수 있는 부분적 복사형상계수 계산을 통한 계산량 절감 기법이다. 이를 통해 기존 방법 대비 커다란 계산 시간의 절감을 이끌어 낼 수 있었다.

## Ⅱ. 본 론

### 2.1 복사형상계수 계산 방법 개요

위성종합설계 소프트웨어에 구현된 복사형상계수 계산법은 기본적으로 직접 수치적분법에 기

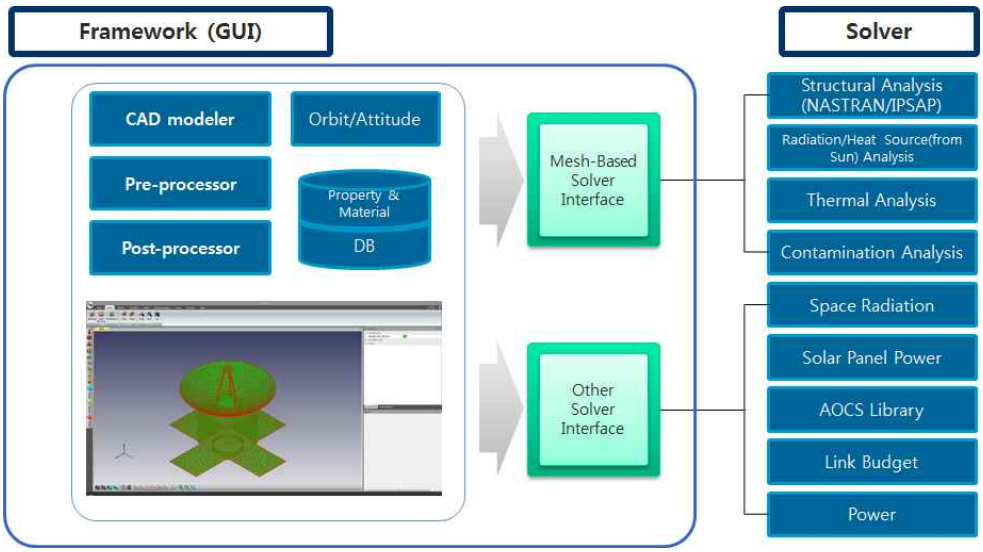


Fig. 1. Overview of KSDS

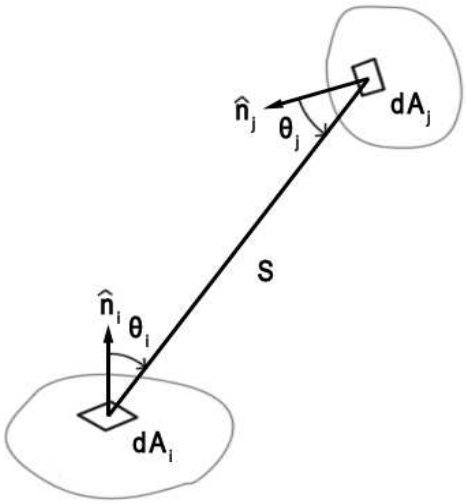


Fig. 2. Radiative Exchange Between Two Infinitesimal Surface Elements

반한다. 복사형상계수의 정의인 Fig. 2와 식 (1)에 따라서 두 요소 간 복사형상계수는 상호성을 가지며, 따라서 이를 수치적으로 이산화하여 나눈 요소들을 식 (2)와 같이 직접 적분할 때는 두 요소의 쌍에 대해서 계산을 수행하므로 전체 요소수의 제곱의 절반의 적분을 수행하게 된다. 그리고 본 프로그램은 삼각형과 사각형으로 이산화된 면적 요소들에 대한 적분을 수행하므로 요소 자체의 복사형상계수는 기본적으로 0으로 취급할 수 있다. 실제로 큰 곡률을 갖는 곡면은 그 자체의 복사형상계수가 0보다 큰 값을 가질 수 있지만, 본 연구에서는 이에 대해 다루지 않는다.

$$F_{ij} = \frac{1}{A_i}\int_{A_i}\int_{A_j}\frac{\cos\theta_i\cos\theta_j}{\pi r_{ij}^2}dS_j dS_i \quad (1)$$
$$= \frac{1}{A_i}\int_{A_i}\int_{A_j}\frac{(\hat{n_i}\cdot\overrightarrow{r_{ij}})(\hat{n_j}\cdot\overrightarrow{r_{ij}})}{\pi r_{ij}^4}dS_j dS_i$$

$$F_{ij} \approx \frac{1}{A_i}\sum_I\sum_J\frac{(\hat{n_I}\cdot\overrightarrow{r_{IJ}})(\hat{n_J}\cdot\overrightarrow{r_{IJ}})}{\pi r_{IJ}^4}\Delta S_J\Delta S_I \quad (2)$$

식 (2)와 같이 적분의 대상이 되는 두 요소를 분할하여 적분을 하기 위하여 식 (3)과 같은 입체각을 기준으로 한 격자 세분화 식을 적용하였다. 이는 가까운 거리의 두 요소의 경우 계산의 정확도를 높일 수 있고 동시에 두 요소가 멀리 떨어져 있어서 복사형상계수의 값이 크지 않을 경우 분할을 적게 함으로서 수치적분에 소요되는 계산 비용을 절감할 수 있는 격자 세분화 기준이다. 본 연구에는 NASA에서 개발한 RAVFAC[21]에서 적용한 크기인 0.01을 적용하고 있으며, 입체각의 크기에 따른 격자 세분화의 적분 결과를 통해[16] 해당 기준이 충분히 엄밀해와의 오차가 1% 이내로 정확성을 만족함을 알 수 있다.

$$\frac{A_i}{S^2} \le \alpha, \quad \frac{A_j}{S^2} \le \alpha \quad (3)$$

## 2.2 복사형상계수 계산 병렬화

복사형상계수의 직접 수치적분은 각 요소의 쌍의 이중 면적분으로 이루어진다. 따라서 이는 쉽게 이중의 반복문으로 구현할 수 있고 각각의 수치적분은 독립적이고 반복문의 전, 후에 영향을 미치지 않는다. 본 연구는 공유메모리 환경하에서 OPENMP[23]를 통해 해석코드를 병렬화하였다.

한편 인공위성의 임무 궤도에 따라 태양전지판과 위성 본체의 상대 변위가 변할 수 있으므로 전체 궤도 주기에서 시간에 따라 변화하는 복사형상계수를 각 시간 스텝 별로 계산할 필요가 있다. 여기서 우리는 병렬화를 위한 방안으로 크게 다섯 가지를 고려할 수 있다[24].

2.2.1는 각 시간스텝 별 독립적인 계산을 병렬적으로 동시에 수행하고, 2.2.2~2.2.4는 각 시간 스텝 별로 내부 루틴인 복사형상계수 계산을 병렬화하고 각 시간 스텝 별 루프는 순차적으로 반복하게 된다. 2.2.5는 2.2.2~2.2.4의 루프 병렬계산의 효율성을 높일 수 있는 동적(Dynamic) 및 안내적(Guided) 부하 조절 기법에 대한 내용이다.

### 2.2.1 시간 스텝 별 병렬계산

각 시간 스텝 별 형상계수는 다른 시간 스텝의 계산과는 무관하게 독립적으로 이루어진다. 따라서 이를 쉽게 각 프로세서에 작업을 할당하여 계산을 수행할 수 있다. 이론적으로는 이 방법이 계산 성능상으로는 가장 빠른 방법이라고 할 수 있다. 다만 할당된 프로세서만큼의 독립된 계산 공간이 필요하다는 단점이 있다. 그리고 2.2.6에서 후술하겠지만 실제 계산 성능은 시간 스텝 내부 작업을 병렬화하는 것과 크게 차이가 나는 것은 아니다.

### 2.2.2 이중 반복문 외부 루프 병렬화

복사형상계수는 요소 간 수치적분이므로 쉽게 이중의 반복문으로 구성할 수 있고 이 가운데 바깥의 반복문을 병렬화하는 방법이다. 여기서 복사형상계수는 그 정의에 의해 면적과의 곱이 대칭이므로 전체 복사형상계수 행렬의 상부 혹은 하부의 절반만 계산을 수행하게 된다. 따라서 외부 루프 부분을 단순히 병렬화할 경우에 각 프로

세서 별 부하 조절이 제대로 이루어지지 않아 병렬성능이 좋지 않을 가능성이 있다. 이를 극복하기 위해 동적 부하 조절 기법 등을 도입하여 해결할 수 있다.

#### 2.2.3 이중 반복문 내부 루프 병렬화

이 방법은 2.2.2과 달리 외부 대신 내부 루프를 병렬화하는 방법이다. 이론상으로는 2.2.2에 비해 비슷한 작업 단위로 이루어진 내부의 루프를 같은 크기의 덩어리로 분할하여 계산하므로 부하 조절 측면에서 보다 유리한 방법이다. 하지만 외부 루프에 의해 이를 순차적으로 반복해서 실행하면 병렬계산의 동기화 등에 보다 많은 추가 비용이 소요되며, 이러한 이유로 일반적으로 외부 루프 병렬화에 비해 내부 루프 병렬화는 그 성능이 좋지 않은 것으로 알려져 있다. 이러한 추가적인 병렬계산 부하로 인해, 참고문헌 [24]와 이후의 2.2.6에서 확인할 수 있듯이 이 방법의 계산 성능은 다른 방법들에 비해 무척 떨어지고 병렬 효율도 좋지 않다.

#### 2.2.4 이중 반복문 통합 단일 루프 병렬화

해당 방법은 2.2.2, 2.2.3과 달리 이중 루프의 작업 내용을 단일의 통합된 루프로 결합하고, 이를 병렬화하게 된다. 복사형상계수 행렬의 상반 혹은 하반의 계산 작업들을 단일의 루프로 결합했을 때 단일 루프 인덱스를 각 행과 열의 인덱스로 추출하는 추가 작업이 필요하다. 이는 단일 루프 인덱스, 각 행과 열의 인덱스 모두 0부터 시작한다고 할 때 식 (4)와 같이 표현할 수 있다. 식 (4)에서 소문자 $i$, $j$는 복사형상계수 행렬의 행과 열의 인덱스이고, 대문자 $I$는 단일화된 루프의 인덱스이다. floor함수는 실수 인자를 받아 그보다 크지 않은 정수 중 최대값을 산출하는 수학함수이다. 본문에서는 복사형상계수 행렬의 상부에 대해 계산을 수행하고, 열의 인덱스는 $i+1$부터 시작한다.

$$i = floor\left(\frac{(2N_e-1)-\sqrt{(2N_e-1)^2-8I}}{2}\right) \qquad (4)$$
$$j = I-\frac{i(2N_e-1-i)}{2}+i+1$$

이론적으로 이 방법은 2.2.2의 외부 루프 별 작업 크기의 불균일성으로 인한 병렬성능 저하를 줄이고, 2.2.3과 같은 내부 루프의 병렬화에 요구되는 추가 비용이 없이 병렬계산을 수행할 수 있는 기법이라고 할 수 있다. 하지만 실제 계산 결과는 2.2.5에서 언급할 작업 부하 조절 기법과 결합하지 않을 때는 2.2.2보다 더 좋지 않을 수도 있다는 것을 2.2.6에서 확인할 수 있다.

#### 2.2.5 반복문 병렬 작업 부하 조절

OPENMP를 활용하여 루프를 병렬화할 때 보통은 각 프로세서 별 작업 크기는 전체 루프 크기를 프로세서 개수로 나눈 수만큼 된다. 예를 들어 10개의 크기를 가진 루프를 2개의 프로세서로 분할할 때, 루프의 1부터 5까지는 1번 프로세서가, 루프의 6부터 10까지는 2번 프로세서가 계산을 담당하게 된다. 이 방법이 정적 스케쥴링(Static Scheduling)으로서 특별한 지시어(Keyword)로 작업 부하 조절을 하지 않을 경우 기본적으로 사용되는 방법이다. 이 방법은 구현이 쉽고 후술할 동적 및 안내적 스케쥴링 기법에 비해 병렬화에 소요되는 추가 비용이 많이 들지 않는다. 하지만 루프 내부의 각 작업 크기가 일정하지 않으면 전체 계산의 종료를 위해 최대 시간이 소요되는 병렬계산을 기다리게 되고 이는 작업 부하 분산 측면에서 바람직하지 않고 많은 유휴시간(Idle Time)이 낭비된다.

이를 극복하기 위해 OPENMP에는 동적, 안내적 작업 스케쥴링 기법을 제공하고 있다. 먼저 동적 스케쥴링은 정적 스케쥴링에서 나오는 유휴시간을 줄이기 위한 기법으로 전체 루프를 Chunk라는 단위로 나눈 후, 이를 계산이 수행되는 상황에 따라 각 Chunk를 작업이 끝난 유휴 프로세서에 할당하는 방법이다. 참고문헌 [24]의 2.3과 유사하나 동적 스케쥴링과의 차이점은 참고문헌 [24]의 2.3은 작업 분배 방식이 이미 고정되어 변하지 않는 반면, 동적 스케쥴링은 계산 진행 상황에 따라 작업 분배가 유연하게 바뀔 수 있다는 점이다. 하지만 Chunk크기에 따라 동적인 스케쥴링에 추가적인 비용이 소요된다.

안내적 스케쥴링은 계산 초기에는 커다란 Chunk로 시작하여, 계산이 진행될수록 Chunk의 크기를 줄여 최종적인 유휴 시간을 줄이고자 하는 기법이다. 안내적 스케쥴링은 동적 스케쥴링에 추가로 필요한 계산 비용을 줄이면서 유휴 시간 최소화를 통해 전체적인 성능 향상을 추구하는 방법이라고 할 수 있다.

이상에서 다룬 병렬화 방법들 2.2.1~2.2.4를 간단한 C 코드 형식으로 나타내면 Table 1과 같다. 2.2.5의 동적, 안내적 기법은 밑줄로 표시된 OPENMP의 for 지시문 우측에 각각 해당 키워드(dynamic, guided)를 추가하는 것으로 쉽게 적용할 수 있다.

#### 2.2.6 병렬해석 결과

제안된 병렬화 방법들을 OPENMP를 활용하여

Table 1. Parallelization Methods of OPENMP

| | |
|---|---|
| 2.2.1 | #pragma omp parallel for<br>for(int t=0;t<ntimes;t++)<br>{<br>for(int i=0;i<nfaces;i++)<br>{<br>for(int j=i+1;j<nfaces;j++)<br>{<br>// numerical integration<br>}<br>}<br>} |
| 2.2.2 | for(int t=0;t<ntimes;t++)<br>{<br>#pragma omp parallel for<br>for(int i=0;i<nfaces;i++)<br>{<br>for(int j=i+1;j<nfaces;j++)<br>{<br>// numerical integration<br>}<br>}<br>} |
| 2.2.3 | for(int t=0;t<ntimes;t++)<br>{<br>for(int i=0;i<nfaces;i++)<br>{<br>#pragma omp parallel for<br>for(int j=i+1;j<nfaces;j++)<br>{<br>// numerical integration<br>}<br>}<br>} |
| 2.2.4 | for(int t=0;t<ntimes;t++)<br>{<br>int npair = (nfaces)*(nfaces-1)/2;<br>#pragma omp parallel for<br>for(int l=0;l<npair;l++)<br>{<br>int i=floor((<br>(2*nfaces-1)-sqrt((2*nfaces-1)*(2*nfaces-1)-8*l))/2);<br>int j= l-i*(2*nfaces-1-i)/2+i+1;<br>// numerical integration<br>}<br>} |
| 비고 | ntimes: 시간 스텝 개수<br>nfaces : 요소 개수<br>npair : (i,j)로 구성된 계산작업 쌍 총 개수<br>i,j : 복사형상계수 행렬의 행과 열 인덱스<br>l : 단일화된 루프의 인덱스 |

개발하였다. 사용된 컴파일러는 인텔 C/C++ 14.0 버전이었고 가장 높은 수준의 코드 최적화 옵션(-O3)을 적용하여 테스트하였다. 사용한 컴퓨터는 8코어 인텔 제온 프로세서(Intel Xeon E7-2830 2.13GHz)를 2개 탑재하여 총 16개의 프로세서로 계산을 수행하였다. 두 개의 문제(Fig. 3, Fig. 8)로 성능을 시험했는데, 첫 번째 문제는

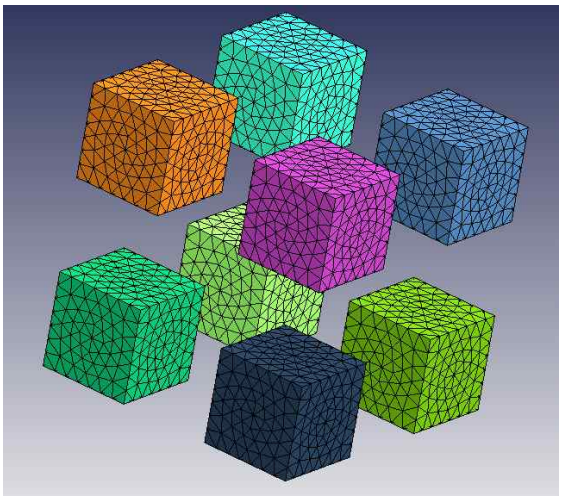

Fig. 3. Tested Problem Visualized by KSDS

Fig. 3와 같이 총 4800개의 요소로 구성된 크기 1의 2x2x2=8개의 정육면체로 이루어진 문제이다. 복사형상계수 계산의 효용성을 보기 위해 충분히 큰 크기의 문제가 필요하기에 각 정육면체의 내부 요소 간 복사형상계수를 계산하는 문제이며 차폐에 의한 성능 차이도 확실히 알 수 있다. 2.3에 언급할 차폐 알고리즘인 KD-Tree 공간 탐색 알고리즘[25-27]으로 병렬계산을 수행하였다.

총 시간 스텝은 16개를 사용하였다. 테스트한 문제의 경우 실제 각 시간 스텝 모두 계산 결과는 동일하지만 이번 논문은 계산 결과가 아닌 계산 성능을 확인하기 위함이므로 모든 시간 스텝마다 동일한 문제를 해석하게 된다.

본 연구는 2.2.1~2.2.4의 네 가지 기본적 병렬화 방법에 더해 2.2.2, 2.2.4의 두 가지 방법의 동적, 안내적 스케줄링 기법을 더한 아래와 같은 총 8가지 기법의 1부터 16개 프로세서의 병렬성능을 확인하였다. 기울임으로 표현된 글자는 이후에 등장할 그래프들에서 표기된 형식이다.

(1) 시간 스텝 별 병렬화(2.2.1) : *시간스텝병렬*
(2) 복사형상계수 행렬 외부 루프 병렬화(2.2.2) : *외부병렬*
(3) 복사형상계수 행렬 외부 루프 병렬화(2.2.2) +동적 스케줄링 : *외부병렬+동적*
(4) 복사형상계수 행렬 외부 루프 병렬화(2.2.2) +안내적 스케줄링 : *외부병렬+안내적*
(5) 복사형상계수 행렬 내부 루프 병렬화(2.2.3) : *내부병렬*
(6) 복사형상계수 행렬 통합 루프 병렬화(2.2.4) : *단일병렬*
(7) 복사형상계수 행렬 통합 루프 병렬화(2.2.4) +동적 스케줄링 : *단일병렬+동적*

(8) 복사형상계수 행렬 통합 루프 병렬화(2.2.4) +안내적 스케쥴링 : *단일병렬+안내적*

위의 8가지 방법에 따른 16개의 프로세서를 사용한 해석 시간은 Fig. 4와 같다. (1)의 결과는 16개의 모든 시간 스텝에 대해 16개의 프로세서를 통해 동시에 수행했을 때의 수치를, 나머지 방법들은 세 개의 시간 스텝의 평균치에 16을 곱한 수치를 적용하였다. 이후에 등장할 모든 계산시간은 1개의 프로세서로 (2)의 방법으로 수행한 계산시간의 비율로 표현하기로 한다.

결과를 보면 (1)이 예상대로 병렬 계산 효율성 면에서 가장 뛰어난 점을 보임을 알 수 있다. 그렇지만 동적 혹은 안내적 스케쥴링이 결합된 (3), (4), (7), (8)의 방법도 (1)과 그리 큰 차이를 보이지 않으면서 메모리 소요량은 그 16분의 1임을 알 수 있기에 계산성능 뿐만 아니라 메모리 소요량 측면에서도 후자의 방법들이 더욱 유용하다고 할 수 있다. 그리고 내부 루프를 병렬화한 (5)는 이미 참고문헌 [24]에 제시된 것처럼 가장 나쁜 효율을 보여준다.

복사형상계수 행렬 루틴을 병렬화하는 방법들인 (2)부터 (8)까지의 7가지 방법들에 대해 프로세서 개수를 1부터 16까지 늘려가며 속도증가 테스트(Speedup Test)를 수행하였다. Fig. 5와 Fig. 6은 각 방법 별 계산시간 비율, Fig. 7은 병렬 속도증가(Parallel Speedup)를 각 방법 별로 프로세서 개수에 따라 나타낸 그림이다. 속도증가(Speedup)는 1개의 프로세서 계산시간을 $n$개의 프로세서의 계산시간으로 나눌 때 나오는 수치를 의미한다.

정적 스케쥴링만 있을 때인 (2), (5), (6)을 보면 (2)가 가장 뛰어난 성능을 보임을 알 수 있다. 이론상으로는 (6)이 다른 방법들에 비해 부하 분

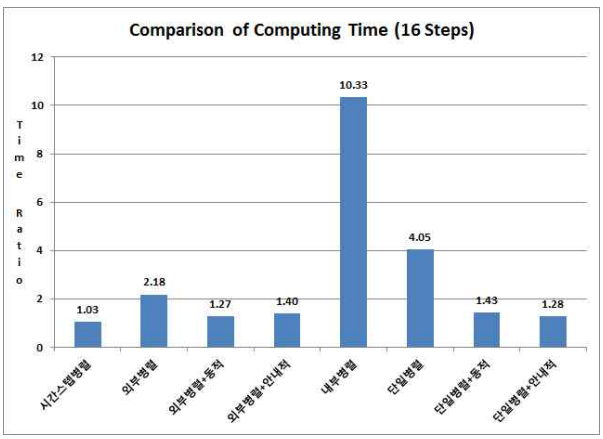


Fig. 4. Comparison of Computing Time of 16 Steps

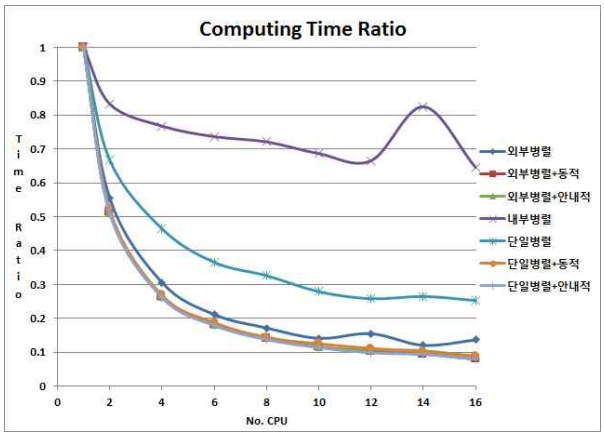


Fig. 5. Computing Time Ratio of (2)~(8)

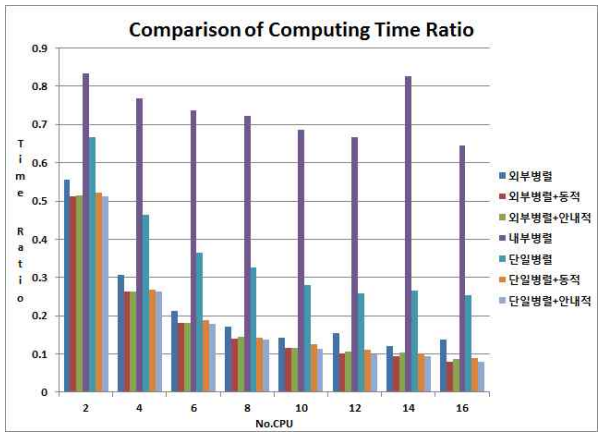


Fig. 6. Comparison of Computing Time Ratio

산에 더 유리한 면이 있어 보이나, 실제 결과는 (2)가 훨씬 뛰어나다. 아마도 이는 내부 차폐 알고리즘 등으로 인한 복사형상계수 각각의 계산량이 동일하지 않은 것으로 인해 실제로는 (6)의 방식으로 작업 분할 시 (2)보다 더욱 불균일하게 분배되었기에 나타난 결과로 생각된다. 한편 (5)는 이미 언급한 데로 프로세서 개수를 증가시켜도 계산 성능이 높아지는 경향이 잘 보이질 않는다. 이러한 점은 이미 참고문헌 [24]에서 예측할 수 있기에 현 연구에서는 (5)의 동적, 안내적 부하 조절 기법을 제외하였다.

그리고 (2), (3), (4) 및 (6), (7), (8)을 각각 비교하면 동적, 안내적 부하 분산 기법이 성능 향상에 큰 기여를 함을 확인할 수 있다. 정적 스케쥴링 작업 부하 조절 시에 단일 병렬 기법 (6)이 예상과는 달리 외부 루프 병렬화 (2)보다 그 성능이 떨어지는 것과 달리 동적/안내적 스케쥴링 기법을 도입하면 양자에 큰 차이를 보이지 않는다. 안내적 스케쥴링 외부 루프 병렬화 (3)과 동적 스케쥴링 단일 병렬 기법 (8)이 가장 뛰어난

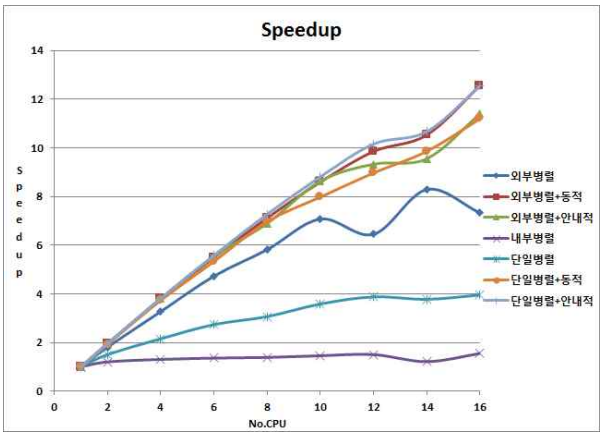


Fig. 7. Parallel Speedup

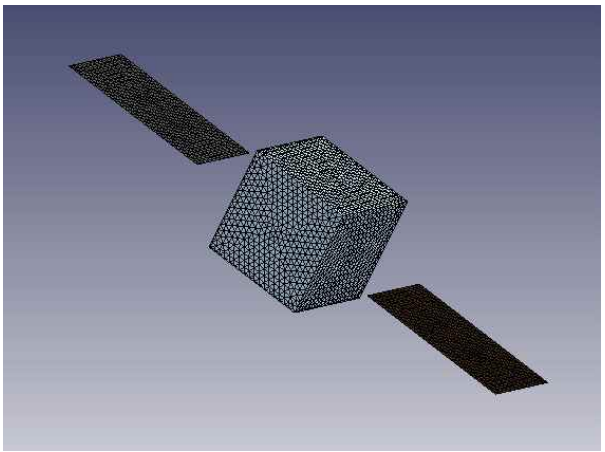

Fig. 8. The Satellite Model with Solar Panels

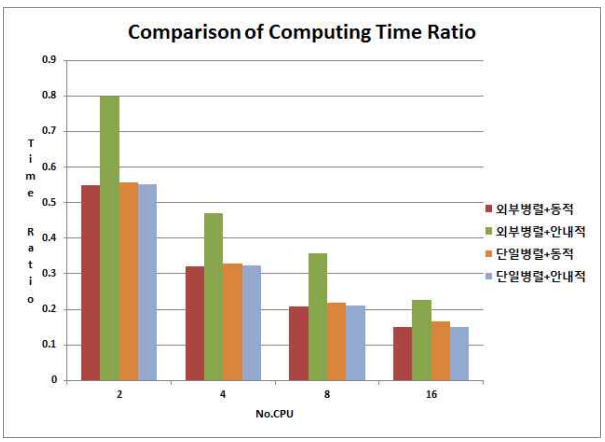


Fig. 9. Comparison of Computing Time Ratio of the Satellite Model

효율을 보이고 있다. 그리고 (3)과 (8), (4)와 (7)이 서로 비슷한 결과를 보인다. 그렇지만 (3), (4), (7), (8)의 네 방법은 Fig. 3의 문제에 있어서는 큰 차이를 보이지 않으며, 기본적으로 동등한 성능을 보여준다고 생각할 수 있다. 그렇지만 다음 예제에는 그 양상이 이것과는 약간 다르며 이후에 언급할 것이다.

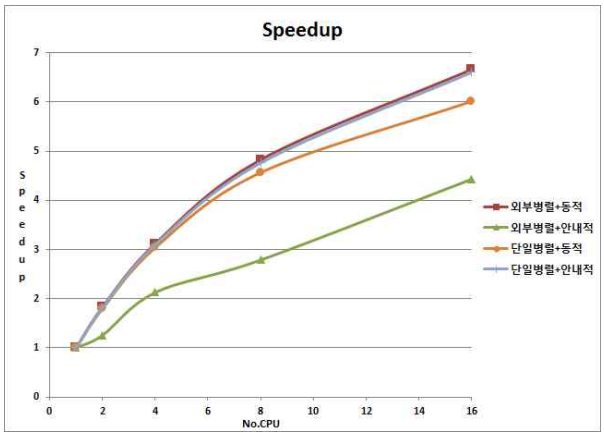


Fig. 10. Parallel Speedup of the Satellite Model

두 번째는 Fig. 8과 같이 이전 예제보다 더 인공위성에 가까운 모델로 검증하였다. 해당 모델은 양 쪽에 두 개의 태양전지판을 가진 정육면체 형상의 본체를 가진 위성 모델로서 본체의 내부는 비어 있고 본체 내, 외부 모두 복사열전달이 일어난다. 따라서 태양전지판과 본체 모두 양면에 복사열전달 요소가 있으며 총 11036개의 요소로 구성되어 있다.

본 예제에서는 이전 예제에서 다른 방법들보다 성능상의 우위를 확인한 (3), (4), (7), (8)에 대해서, 프로세서 개수를 2,4,8,16개로 늘려가며 성능 시험 비교를 수행하였다. Fig. 9에 실행 시간 비율을, Fig. 10에 속도증가(Speedup)를 나타내었다. 현 예제도 이전과 비슷하게 (3), (8)이 양자가 비슷한 정도로 (4), (7)보다 뛰어난 성능을 보이며, (4)가 이 중에서는 눈에 띌 정도로 가장 좋지 않은 성능을 보인다.

한편 이전 예제의 속도증가 그래프 Fig. 7과 Fig. 10을 비교하면 전자의 문제(Fig. 7)의 병렬성능이 훨씬 더 좋음을 알 수 있다. 이는 현재 연구에서는 KD-Tree의 구축은 병렬화가 되어 있지 않았고, 전자의 문제는 차폐 탐색을 위한 KD-Tree 구축 시에 드는 계산이 복사형상계수 자체의 수치적분보다 훨씬 작은 반면 후자는 상대적으로 그 비율 차이가 크지 않기 때문이다.

두 예제를 종합하면, (1)의 시간 스텝 병렬화가 계산 시간 측면에서 가장 큰 이점을 볼 수 있으나 요구하는 메모리 공간이 그만큼 크다는 점이 단점이다. 외부 루프를 동적 스케쥴링 기법으로 병렬화한 (3)과 단일화된 통합 루프를 안내적 스케쥴링 기법으로 병렬화한 (8)이 비슷한 성능을

보여줄 수 있지만, 실제 코드 구현 측면에서 (3)이 (8)보다 쉽기에 (3)의 방법이 가장 추천할 수 있는 병렬화 기법이라고 할 수 있다.

## 2.3 KD-Tree 차폐 탐색 알고리즘

복사형상계수 계산을 위해서는 두 요소의 임의의 두 절점을 직선으로 이었을 때 차폐 여부를 확인해야 한다. 기존 방식은 모든 요소에 대해 차폐 여부를 검사하는 방식으로서 이러한 방법의 계산량은 하나의 요소 쌍의 탐색에 대해서 $O(N)$이므로 전체 계산량은 $O(N^3)$이 된다. 여기서 $N$은 전체 요소의 수이다.

KD-Tree는 공간상에서 이진 트리(Binary Tree) 방식으로 절점을 포함한 삼각형, 사각형 등의 도형 및 그래픽 객체를 저장하고, 이를 통해 광선 차폐 등의 탐색에 활용하는 공간 탐색 기법이다. 만일 모든 객체들이 완전 이진 트리(Complete Binary Tree)에 저장될 때, 단일 탐색에 소요되는 계산량은 $O(logN)$이므로 전체 계산량은 $O(N^2logN)$이다.

본 연구에 적용된 KD-Tree 광선추적 알고리즘은 참고문헌 [25-27]에 구현된 것을 본 연구에 맞게 이식(Porting)하였다. 해당 알고리즘은 광선추적에 더욱 최적화하기 위해 완전 이진 트리 방식 대신 독자적인 SAH(Surface Area Heuristic)라는 기준으로 공간을 이분할한 후 분할된 공간에 걸치는(Intersection) 객체들을 각 공간에 해당하는 하부 트리에 포함시킨다. 최적의 공간 분할을 찾기 위해 SAH 개념으로 정의한 비용함수를 최소화하는 절단면의 축과 좌표값을 계산한다. 현재는 KD-Tree의 구축에는 병렬화가 적용되어 있지 않고 복사형상계수 병렬 계산 내부에서 쓰레드 안전(Thread-Safe)한 탐색 함수를 호출하고 있지만, KD-Tree의 구축은 복사형상계수 계산에 비해 상대적으로 그 비중이 작기에 본 연구에서는 이 부분은 생략하도록 한다.

본 연구에 구현된 KD-Tree 기반 차폐 탐색 알고리즘의 계산 성능과 기존의 일대일 탐색 방식과의 비교는 Fig. 11과 같다. 2.2.6에서 언급된 두 문제(Fig. 3, Fig. 8)에 대해 각각 KD-Tree 차폐 알고리즘의 계산 시간 대비 일대일 차폐 탐색의 계산 시간의 비율의 그래프를 Fig. 11에 표현하였다. 정육면체 문제(Fig. 3)는 약 35배, 위성체 모델(Fig. 8)은 약 7.62배의 계산 성능 차이를 보인다. 이 결과는 실제 계산으로 산출한 것이므로 문제마다 그 비율은 다를 수 있지만, 일반적으로 큰 문제일수록 KD-Tree 기반 차폐 탐색 알고리즘이 더 큰 효율성을 보임을 유추할 수 있다.

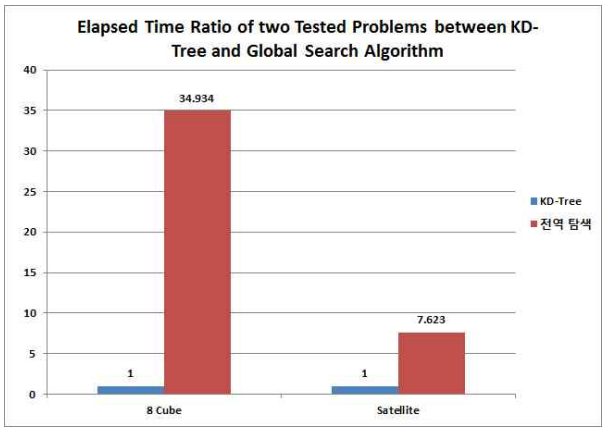


Fig. 11. Elapsed Time Ratio of Two Tested Problems between KD-Tree and Global Search Algorithm

## 2.4 부분 복사형상계수 갱신

인공위성은 그 임무에 맞게 정해진 궤도를 운행한다. 보통 본체는 통신 및 촬영 등을 위해 지구를 지향하고, 태양전지판은 발전을 위해 태양을 지향하는 경우가 많기에 한 궤도상에서도 그 위치에 따라 본체와 태양전지판의 상대적 변위가 존재하게 된다. 이럴 경우 본체와 태양전지판 사이의 복사형상계수 역시 변하므로 다시 계산할 필요가 있다.

이러한 현상을 효율적으로 처리하기 위해 본 논문에서는 부분 복사형상계수 행렬 갱신(Updating Partial View Factor Matrix)라는 개념을 고안하였다. 이 방법은 이전 시간 스텝과의 상대적 변위 차이를 확인하여 상대적 변위가 발생한 요소들에 대해서만 복사형상계수를 재계산하는 방법이다. 해당 방법은 재계산 대상이 되는 요소와 나머지 요소들 사이의 상대적 움직임이 크지 않아 차폐에 아무런 영향을 미치지 않을 때 유용하게 적용할 수 있다.

인공위성의 태양전지판과 본체의 상대적 운동이 있더라도 서로의 차폐에는 영향을 미치지 않는다. 만일 태양전지판이 본체 내부로 침입하게 되면 본체 내부 요소 간 복사형상계수 계산에도 태양전지판이 영향을 미치게 되므로 이럴 때는 전체 형상에 대해 복사형상계수를 계산하여야 한다. 그렇지만 인공위성의 열해석에 이러한 극단적인 경우를 가정할 필요가 없기에 태양전지판과 본체 사이의 복사형상계수만 갱신하면 된다. 이러한 이유로 부분 복사형상계수 행렬 갱신이라는 명칭을 붙였다.

해당 개념을 검증하기 위해 2.2.6절에 소개한 Fig. 8의 문제를 부분 복사형상계수 행렬 갱신

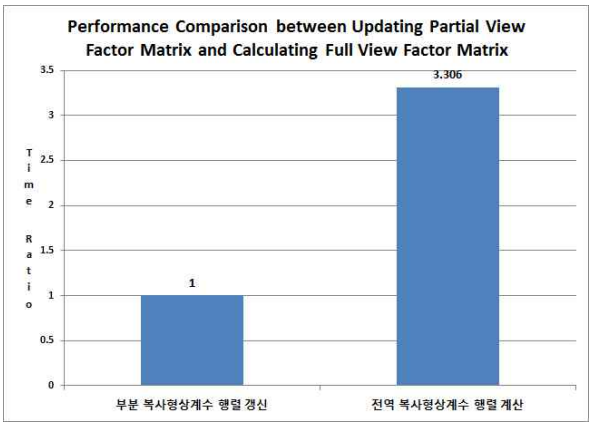


Fig. 12. Performance Comparison between Updating Partial View Factor Matrix and Calculating Full View Factor Matrix

기법을 사용한 것과 그렇지 않고 전체 요소의 복사형상계수를 계산하는 기존 방식과의 성능 비교를 수행하였다. Fig. 12의 결과를 보면 알 수 있는 것처럼 부분 복사형상계수 행렬 갱신 기법이 전체 복사형상계수 계산보다 약 3.3배 정도 계산 시간을 절감할 수 있음을 알 수 있다.

2.2절의 동적 외부 루프 병렬화 기법으로 16개의 CPU를 활용하여 KD-Tree 차폐 탐색, 부분 복사형상계수 갱신 기법으로 Fig. 8의 위성 문제를 해석한 경우와 해당 기법이 전혀 적용되지 않은 채 1개의 CPU로 계산을 수행하였을 때의 계산 시간을 비교하니 약 650배 가량의 성능향상을 관측할 수 있었다. 이를 통해 본지에서 제시한 기법들을 통해 복사형상계수 계산에 무척 큰 성능 향상을 얻을 수 있음을 알 수 있다.

## III. 결 론

본 연구에서는 복사형상계수의 효율적인 계산을 위한 다중프로세서 병렬화, KD-Tree 차폐 탐색 알고리즘, 부분 복사형상계수 갱신의 세 가지 기법들을 소개하고 그 실측 성능을 보여줌으로서 그 효용성을 보였다. 현대의 다중코어 컴퓨팅 환경에서는 병렬성이 필수적이므로 본지에서 제안한 병렬화 방법 중 외부 루프 동적 스케쥴링 병렬화가 성능상으로 가장 뛰어나면서도 알고리즘 구현이 다른 병렬화에 비해 용이하다는 장점이 있기에 복사형상계수 병렬화에 가장 뛰어난 방법이라고 할 수 있다. 그리고 차폐 탐색 알고리즘으로 KD-Tree를 도입하고, 위성 궤도 상 상대변위가 발생하는 문제에 대해 부분 복사형상계수 갱신 기법을 통해 계산성능을 수배 이상 향상시킬 수 있다. 요소가 수천~수만 개 수준으로 갈수록 복사형상계수 계산에도 많은 시간이 소요되기에 본지에서 제안하고 검증한 방법론의 적용이 필수적이라고 할 수 있다.

## References

1) Y.H.Kim, I.H.Choi, Y.H.Jeon, B.S.Hyun, H.Y.Jeon and J.H.Kim, “Functions of Pre/Post Processor using Finite Element Method,” KSAS 2014 Spring Conference, Apr. 2014, pp. 944-947.

2) S.C.Lee, M.K.Kim, H.D.Kim, D.Y.Ryu, E.S.Sim, S.H.Yoon, and Y.H.Kim, “Development of Attitude Control Module for KARI Satellite Design Software,” KSAS 2015 Spring Conference, Apr. 2015, pp. 787-791.

3) M.K.Kim, B.S.Hyun, J.H.Kim, J.M.Woo, J.Y.Cho, “ Functionalities and Verification of the Satellite Thermal Analysis Solver Based on Finite Element Method,” KSAS 2014 Spring Conference, Apr. 2014, pp. 948-951.

4) J.J.Lee, B.S.Hyun, J.H.Choi and T.K.Kim, “Introduction and Verification about Solver of View Factor and External Heat Source in KSDS,” KSAS 2014 Spring Conference, pp. 952-955.

5) C.H.Lee, J.H.Lee and Y.S.Cheon, “Introduction of KSDS Contamination Analysis and Its Verification,” KSAS 2014 Spring Conference, Apr. 2014, pp. 956-959.

6) K.J.Park, Y.Y.Park, J.R,Lim and H.T.Choi, “AOCS Library Development for KARI Satellite Design System,” KSAS 2014 Spring Conference, Apr. 2014, pp. 1004-1007.

7) Y.J.Cho, S.K.Lee and J.H.Seon, “Development and Verification of Calculation Program for Total Dose and Solar Cell Degradation by Space Radiation,” KSAS 2014 Spring Conference, Apr. 2014, pp. 1008-1011.

8) W.K.Lim, S.B.Ryu, J.P.Kim, S.I.Lee, S.K.Kim and S.K.Lee, “KSDS S Band Communication Link Analysis Solver Function and Verification for Low Earth Orbit and Geostationary Orbit,” KSAS 2014 Spring Conference, Apr. 2014, pp. 1012-1015.

9) J.C.Koo and J.B.Jang, “Function and Verification of the Power Analysis Solver in

KARI Satellite Design System," KSAS 2014 Spring Conference, Apr. 2014, pp. 1016-1020.

10) http://ipsap.snu.ac.kr

11) K.J.Park, Y.J.Park, J.Y.Cho, C.Y.Park and S.J.Kim, "Design Optimization of a Wing Structure under Multi Load Spectra using PSO algorithm," Journal of KSAS, Vol. 40, No. 11, Nov. 2012, pp. 963-971.

12) M.K.Kim, J.H.Kim, C.Y.Park and S.J.Kim, "Parallelization of Multifrontal Solution Method for Shared Memory Architecture," Journal of KSAS, Vol. 40, No. 11, Nov. 2012, pp. 972-978.

13) M.K.Kim and S.J.Kim, "An Out of Core Linear Direct Solution Method for Large Scale Structural Analysis," Journal of KSAS, Vol. 42, No. 6, Jun. 2014, pp. 445-452.

14) M.K.Kim, "A Study of Temperature Transfer Algorithm of Distinguished Grids between Thermal and Structural Mesh for Satellite Design", Journal of KSAS,, Vol. 43, No. 9, Sep. 2015, pp. 805-813.

15) D.K.Kim, K.Y.Han, J.H.Choi, J.J.Lee and T.K.Kim, "Study on Radiation View Factor Calculation by Mesh Subdivision Method Considering Solid Angle", KSAS 2013 Fall Conference, Nov. 2013, pp. 745-748.

16) Satellite System Virtual Design Software Development, KARI Research Report, 2013, pp. 33-41.

17) J.H.Choi, J.H.Kim, I.H.Jung, P.H.Lee, and T.K.Kim, "Study on IR Signature Characteristics for different Transmittance over the Korean South Sea during Summer and Winter Seasons," Journal of the Korea Institute of Military Science and Technology, Vol. 13, No. 2, Apr. 2010, pp. 320-327.

18) Gilmore, D.G., "Spacecraft Thermal Control Handbook," American Institute of Aeronautics and Astronautics, Vol. 1, 2002, pp. 36-47.

19) Allen, R.C., Keith, R.J. and Eric, A.M., 1995, "Automated Radiation Modeling for Vehicle Thermal Management," SAE International Congress and Exposition, 1995.

20) Radtherm IR, ThermoAnalytics, http://www.thermoanalytics.com

21) Lovin, J.K., Lubkowitz, A.W., "User's Manual for RAVFAC", NASA Contractor Report, 1969.

22) Modest, M.F., "Radiative Heat Transfer", Academic Press, 2003, pp. 762-778.

23) http://www.openmp.org

24) M.K.Kim, "Parallelization and Performance Enhancement of Radiation Exchange Coefficient Calculation Program in the KARI Satellite Design Software" KSAS 2014 Fall Conference, Nov. 2014, pp. 1528-1531.

25) http://www.flipcode.org/archives/Raytracing_Topics_Techniques-Part_7_Kd-Trees_and_More_Speed.shtml

26) http://www.flipcode.org/archives/raytracer7.zip

27) Ingo Wald, "Realtime Ray Tracing and Interactive Global Illumination", Ph.D Thesis, Saarland University, 2004.