THE JOURNAL OF KOREAN INSTITUTE OF ELECTROMAGNETIC ENGINEERING AND SCIENCE. 2020 Mar.; 31(3), 245～252.

http://dx.doi.org/10.5515/KJKIEES.2020.31.3.245
ISSN 1226-3133 (Print) · ISSN 2288-226X (Online)

GPU 기반의 TSM-RT를 이용한 풍력 블레이드의 동적 RCS 계산

Dynamic RCS Calculation of Wind Turbine Blade Using GPU-Based TSM-RT

최 영 재 · 최 인 식

Young-Jae Choi · In-Sik Choi

요 약

다수의 풍력 블레이드에 의하여 발생하는 동적 RCS를 계산하기 위해서는 샘플링 시간별로 이동된 표적의 3차원 메쉬에 대한 RCS를 계산하는 과정이 필요하다. 본 논문에서는 물리광학법을 사용하여 이러한 동적 RCS를 계산하는 방법을 제안하였다. 광선추적법을 가속하기 위하여 GPU 기반의 TSM-RT 방법을 적용하여 계산 속도를 향상시켰으며, 제안한 방법의 성능을 상용 전자파 해석 소프트웨어인 FEKO와 비교하여 검증하였다. 마지막으로 제안된 방법을 사용하여 서로 다른 회전 속도를 갖는 두 개의 풍력 블레이드의 동적 RCS를 계산하였다.

Abstract

It is necessary to calculate the RCS generated by several wind blades for the three-dimensional mesh analysis of a target moved for each sampling time. In this paper, we propose a method of calculating dynamic RCS through physical optics. The computational speed was improved by applying the graphics processing unit-based two-scale-model ray-tracing method to enhance the ray tracing efficiency. Finally, the proposed method was used to calculate the dynamic RCS of two wind blades at different rotational speeds.

Key words: Dynamic Radar Cross Section, Ray Tracing, Physical Optics, Octree, Two Scale Model, Wind Turbine Blade

Ⅰ. 서 론

최근 레이다 신호처리 분야에서 동적 RCS(Radar Cross Section)를 이용하는 연구들이 많은 관심을 받고 있다. 이러한 연구들의 주된 응용분야는 풍력 발전기의 검출[1]~[3], 드론 검출[4]~[6], 헬리콥터 검출[7]~[9] 등으로 다양하다. 레이다를 이용하여 풍력 발전기를 관측하여 얻은 동적 RCS는 블레이드의 거동에 대한 정보를 가지고 있으므로 이러한 정보를 추출하면, 원격지에서 블레이드의 동작 상태를 추정하는 것이 가능하다. 이러한 연구를 위해선 회전하는 풍력 블레이드의 동적 RCS를 계산할 수 있어야 한다. 그러나 현재까지 수행된 대부분의 동적 RCS 관련 연구들은 수학적인 모델링을 통하여 동적 RCS를 계산하거나[10], 표적의 회전에 대응하는 관측 각도를 역산하여 수치해석 소프트웨어에 대입하는 간접적인 방법을 사용하여 동적 RCS를 계산하였다[11]. 수학적인 모델링을 이용한 동적 RCS의 계산 방법은 빠른 계산이 가능하고, 블레이드의 회전에 의한 성분을 정확히 계산할 수 있다. 그러나 블레이드의 형상에 의한 RCS의 변화를 반영하지 못하므로 진폭에서 실제 결과와 큰 차이를 가지게 된다. 반면,

「본 연구는 한국전력공사의 사외공모 기초연구(개별)에 의해 지원되었음(과제번호: R19XO01-50).」
「이 논문은 2018년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2018R1D1A1B07041496).」
한남대학교 전기전자공학과(Department of Electrical and Electronic Engineering, Hannam University)
· Manuscript received December 16, 2019 ; Revised February 21, 2020 ; Accepted March 21, 2020. (ID No.20191216-135)
· Corresponding Author: In-Sik Choi (e-mail: recog@hnu.kr)

블레이드의 회전에 따른 관측 각도를 역산하는 방법은 3차원 CAD(Computer-Aided Design) 모델로부터 RCS를 계산함으로 표적의 형상에 의한 RCS의 변화를 정확히 반영할 수 있다. 그러나 이러한 방법은 다수의 블레이드에 의하여 발생하는 동적 RCS는 계산할 수 없다는 한계를 지니고 있다.

이러한 한계점들을 극복하고, 수치해석 방법을 사용하여 다수의 풍력 블레이드의 동적 RCS를 계산하기 위해서는 메쉬를 사용하여 각 시간 샘플에서의 3차원 장면을 표현하고, 전자파 수치해석 방법으로 이 장면 전체에 대한 RCS를 계산하는 방법을 사용해야 한다. 이러한 방법을 구현하기 위해서는 크게 두 가지 중요한 부분이 있다. 한 가지는 움직이는 3차원 장면의 메쉬를 생성해야 한다는 것이다. 다른 한 가지는 이러한 동적 메쉬에 대하여 1초 동안 관측되는 동적 RCS를 계산하기 위해서는 1초간의 샘플링 횟수만큼 RCS 계산을 반복해야 한다는 점이다. 이것은 연구에 사용 가능한 수준의 동적 RCS 계산 속도를 제공하기 위해선 RCS의 계산 속도가 빨라야 함을 의미한다.

본 논문에서는 앞에서 언급한 문제들의 해결 방법을 소개하였다. 그리고 소개한 방법들을 이용한 동적 RCS 시뮬레이션 방법을 제안하였다. 제안한 방법의 정확성과 계산 속도를 상용 전자파 수치해석 소프트웨어인 FEKO와 비교하여 검증하였다. 마지막으로 제안한 방법을 사용하여 회전하는 두 개의 풍력 블레이드에 대한 동적 RCS를 계산한 결과를 제시하였다.

Ⅱ. 동적 RCS 시뮬레이터

2-1 동적 RCS의 계산 방법

동적 RCS는 움직이는 표적의 시간에 따른 RCS의 변화를 의미한다[12]. 움직이는 표적의 RCS는 식 (1)과 같이 표현할 수 있다.

$$E_p = \sum_{m=1}^{M} A_m(t_p)\exp\{-j4\pi f R_m(t_p)/c\} \tag{1}$$

여기서 t_p는 p번째 시간 샘플에서의 시간을, f는 입사파의 주파수를, 전체 장면을 구성하는 M은 산란점의 개수를 $A_m(t_p)$는 p번째 시간 샘플에서 m번째 산란점의 복소 진폭을, $R_m(t_p)$는 p번째 시간 샘플에서 m번째 산란점과 안테나 사이의 거리를, $c=3\times10^8$ m/s는 진공에서의 빛의 속력을 의미한다.

문제를 단순화하기 위하여 '레이다에서 방사된 전자파가 표적을 맞고 되돌아오는데 걸리는 시간은 같은 시간 동안의 표적의 움직임을 무시할 수 있을 정도로 충분히 짧다'고 가정한다. 따라서 각 시간 샘플 사이의 간격 $\triangle t = t_{p+1} - t_p$는 식 (2)와 같이 나타낼 수 있다.

$$\triangle t = \frac{1}{PRF} + \frac{2R}{c} \approx \frac{1}{PRF} \tag{2}$$

여기서 R은 표적과 안테나 사이의 거리이고, PRF(Pulse Repetition Frequency)는 펄스반복주파수이다.

그림 1은 전자파 수치해석 방법을 이용한 동적 RCS 계산의 개념을 보여준다.

그림 1에서는 각 시간 샘플에 대한 3D 메쉬를 생성하고, 이에 대한 RCS를 계산하는 방법으로 동적 RCS를 획득한다. 이러한 개념으로 동적 RCS를 계산하기 위해서는 모든 시간 샘플에 대한 3D 메쉬가 필요하다. 그러나 1 kHz의 PRF를 가정할 경우, 1초의 동적 RCS를 얻기 위해 필요한 3D 메쉬의 개수는 1,000개에 달하므로 동적 RCS 계산에 필요한 3D 메쉬를 사용자가 모두 작성하는 것은 어렵다. 또한 1초의 동적 RCS를 얻기 위하여 1,000회의 RCS 계산을 수행해야 하므로 RCS 계산에 필요한 시간이

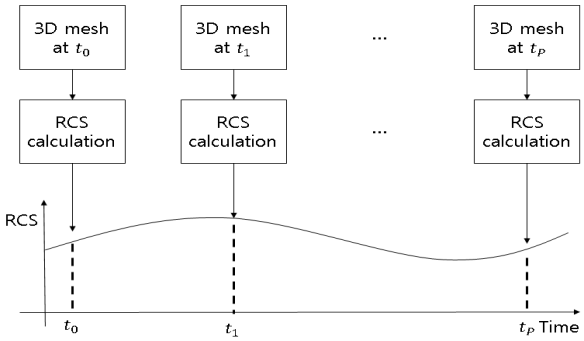


그림 1. 전자파 수치해석 방법을 이용한 동적 RCS 계산 방법의 개념

Fig. 1. Concept of dynamic RCS calculation using numerical method.

극단적으로 길어짐을 알 수 있다.

2-2 움직이는 3차원 메쉬 생성

블레이드의 움직임을 표현하기 위해서 블레이드를 강체(rigid body)로 가정한다. 블레이드의 운동을 표현하기 위해서는 블레이드의 초기 위치와 회전 운동에 의한 위치의 변화를 블레이드의 3차원 메쉬에 반영해야 한다. 이것은 각 블레이드의 메쉬에 회전변환과 이동변환을 적용함으로써 표현이 가능하다. 그림 2는 움직이는 3차원 메쉬를 생성하는 절차를 보여준다.

메쉬의 회전 변환은 식 (3)과 같이 계산이 가능하다.

$$\overrightarrow{V_{rotated}} = \overline{R}\,\overrightarrow{V} \tag{3}$$

여기서 $\overrightarrow{V} \in \mathbb{R}^3$는 메쉬를 이루는 정점(vertex)의 위치벡터이고, $\overline{R} \in \mathbb{R}^{3\times 3}$은 3차원 공간에서의 오일러 회전 행렬이다[13].

메쉬의 이동 변환은 식 (4)와 같이 계산이 가능하다.

$$\overrightarrow{V_{translated}} = \overrightarrow{V} + \overrightarrow{M} \tag{4}$$

여기서 $\overrightarrow{M} \in \mathbb{R}^3$은 메쉬의 이동을 위한 벡터이다.

메쉬의 회전 변환과 이동변환은 동일한 연산을 메쉬를 이루는 수 많은 정점에 대해 수행하는 것이다. GPU를 이용하여 병렬화 된 선형대수 계산은 이미 많은 라이브러리들이 개발되어 있기 때문에 이러한 라이브러리를 사용하는 것만으로도 간단하게 GPU 기반의 병렬화가 가능하다. 본 논문에서는 NVIDIA에서 제공하는 GPU 기반의 선형대수 계산 라이브러리인 CUBLAS를 사용하여 해당 연산들을 구현하였다[14]. 앞에서 설명한 방법으로 움직이는 3차원 메쉬를 표현하기 위해서는 각 부분의 메쉬에 회전변환과 이동 변환을 어떠한 순서로 적용하고, 전자파 수치해석을 수행할 것인지를 지시하는 스크립트(script)가 필요하며, 이러한 스크립트는 변수, 함수, 조건문, 반복문과 같은 최소한의 문법을 지원해야 한다. 본 논문에서는 수치해석 코드에 프로그래밍 언어인 Lua의 인터프리터를 내장(embedding)하여 이 문제를 해결하였다[15]. 따라서 3차원 메쉬의 움직임은 표 1과 같이 Lua 문법으로 표현이 가능하다.

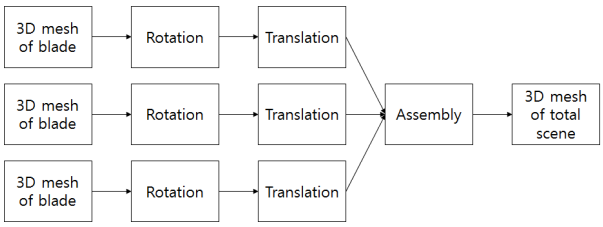


그림 2. 움직이는 3차원 메쉬를 생성하는 절차
Fig. 2. Procedure to create a moving 3D mesh.

표 1과 같은 스크립트를 이용하여 그림 2와 같은 절차를 표현할 수 있으며, 이러한 반복문에 표 1의 11번 줄과 같이 RCS 해석 함수를 배치하여 해석 결과를 배열에 저장하는 방식으로 움직이는 표적의 표현과 이에 대한 동적 RCS의 계산을 모두 수행할 수 있다.

2-3 GPU 기반의 광선추적법 가속

동적 RCS를 계산하기 위해서는 레이다의 PRF와 관측시간을 곱한 횟수만큼의 RCS 계산이 필요하다. 물리광학법은 높은 주파수에서 RCS를 해석하는 기법들 중에서도 계산속도가 매우 빠른 방법이다. 물리광학법에 의하여 평면파가 입사하는 표면의 전류는 식 (5)와 같이 나타낼 수 있다.

표 1. 동적 메쉬의 표현을 위한 Lua script의 예시
Table 1. Example Lua script to implement a dynamic mesh.

Line	Code	Comment
1	for p=0, 1,000 do	Start of loop
2	mesh_add(mesh 1)	Add part to id=1
3	mesh_add(mesh 2)	Add part to id=2
4	id=1	-
5	mesh_rotate(id, 'z', rot_angle)	Rotate mesh of id=1
6	mesh_move(id, x, y, z)	Translate mesh of id=1
7	id=2	-
8	mesh_rotate(id, 'z', rot_angle)	Rotate mesh of id=2
9	mesh_move(id, x, y, z)	Translate mesh of id=2
10	total_mesh=assembly()	Assemble the mesh of all parts.
11	E(p)=calc_rcs(total_mesh)	Calculate rcs of total mesh
12	end	End of loop

$$\vec{J}=\begin{cases}2\vec{n}\times\vec{H_i}, & lit\,region\\ 0 & , shadow\,region\end{cases} \quad (5)$$

여기서 $\vec{H_i}\in\mathbb{C}^3$는 입사파의 자계이고, $\vec{n}\in\mathbb{R}^3$은 평면파가 입사하는 표면의 법선 벡터이다.

식 (5)에서 $\vec{J}\in\mathbb{C}^3$는 입사파가 직접 도달하는 전면(lit region)과 도달하지 못하는 은면(shadow region)으로 나누어 계산하게 된다. 따라서 물리광학법으로 전류를 계산하기 전 은면을 미리 제거하는 과정이 필요하다. 이러한 은면 제거 과정은 물리광학법에 의한 RCS 계산 과정에서 가장 많은 계산시간을 요구하는 과정이다. 따라서 물리광학법의 계산 속도를 향상시키기 위해서는 반드시 광선추적법의 계산 속도를 향상시켜야만 한다. 입사파가 정의되었을 때, 은면을 제거하는 방법은 내적에 의한 방법, Z-buffer 방법, 광선추적법 등 많은 방법이 알려져 있으며, 최근에는 광선추적법 기반의 방법이 많이 사용된다[16]. 물리광학법의 구현에서 광선추적법은 메쉬를 구성하는 각각의 삼각형 패치에 입사하는 광선들이 다른 삼각형 패치에 가로막히지 않고 도달할 수 있는 경우, 해당 삼각형 패치를 전면으로 그렇지 않은 경우는 은면으로 처리하는 방식이다. 이것을 그냥 계산할 경우, 하나의 삼각형 패치가 은면인지를 판단하기 위하여 메쉬를 구성하는 모든 삼각형 패치에 대한 충돌 검사를 수행해야 하고, 이는 매우 많은 계산량을 요구한다. 이러한 문제점을 개선하기 위하여 메쉬가 존재하는 공간을 분할하여, 충돌 계산시 공간과의 충돌을 우선 계산하고, 해당 공간에 위치한 삼각형 패치들만을 충돌계산에 참가시키는 방식의 가속 방법을 사용한다. 이러한 가속법의 대표적인 예시는 octree, kd-tree, bvh-tree 등이 알려져 있다[17]~[19]. 또한 이러한 광선추적법을 GPU 기반으로 병렬화시킴으로써 계산 속도를 향상시키는 연구들 또한 활발하게 이루어졌다[18],[19]. 본 논문에서는 NVIDIA의 GPU 컴퓨팅 라이브러리인 CUDA를 사용하여 GPU 기반의 octree를 구현하였다.

GPU를 이용하여 octree 기반의 광선추적법을 구현할 때의 어려운 점은 포인터에 의한 메모리 연결 구조와 연결리스트로 구성된 octree를 GPU로 전달하는 것이 어렵다는 것이다. 이러한 문제를 해결하기 위하여 그림 3과 같이 CPU에서 octree를 생성한 뒤, 포인터 대신 배열의 인덱스를 이용하여 octree의 각 노드를 방문할 수 있도록

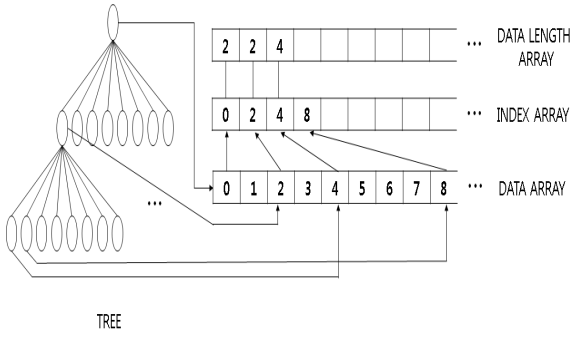


그림 3. Octree를 배열 형태로 바꾸는 방법
Fig. 3. Method to convert an octree into an array form.

한 배열의 형태로 전환하였다. 이러한 방법을 사용하여 성능저하 없이 GPU에서 동작하는 octree 기반의 광선추적법을 구현할 수 있다.

그림 3에서 octree의 각 노드는 DATA ARRAY 배열에 깊이 우선 탐색의 방문 순서대로 저장된다. 이때, 각 노드 데이터가 시작되는 지점과 각 노드의 길이를 각각 INDEX ARRAY 배열과 DATA LENGTH ARRAY 배열에 저장한다. 이러한 정보들을 이용하여 DATA ARRAY로부터 원하는 노드의 정보를 추출할 수 있다.

2-4 TSM-RT을 이용한 광선추적법 가속

물리광학법을 계산할 때, 사용되는 삼각형 패치 한 변의 길이는 $\lambda/8$가 권장된다. 이는 각 삼각형 패치에서 발생하는 전계를 적분하기 위하여 필요한 조건이다. 따라서 이 조건은 주파수에 의존한다. 그러나 광선추적법에서 필요한 삼각형 패치는 3차원 모델의 경계를 정확하게 표현하는 것이 주된 역할이다. 따라서 광선추적법에서 요구되는 삼각형 패치의 크기 조건은 표적의 형상의 복잡도와 표면의 곡률과 같은 요소에 영향을 받을 뿐 주파수와는 무관하다. 따라서 광선추적법에서 요구되는 조건의 삼각형 패치를 사용하여 octree를 생성하고, 생성된 octree를 사용하여 기존의 $\lambda/8$ 조건을 만족하는 메쉬의 충돌 검사를 수행하는 방법으로 은면 처리에 필요한 연산량을 크게 줄일 수 있다. 이러한 방법을 Two scale model ray tracing (TSM-RT)이라 한다[20]. 추가로 물리광학법에서 단일 반사만을 고려할 경우, 입사광선 $\vec{s}$와 삼각형 패치의 법선벡터 $\hat{n}$이 $\vec{s}\cdot\hat{n}\le 0$인 경우, octree를 구성하는 메

쉬에서 제외해도 무방하다. 이러한 방법으로 TSM-RT 방법의 연산량을 감소시킬 수 있다. 본 논문에서는 이 두 가지 방법을 모두 적용하였다.

2-5 GPU 기반의 전계 적분 가속

완전 도체를 가정할 경우, 앞에서 설명한 방법으로 삼각형 패치 표면의 전류를 구하고 나면 식 (6)과 같은 원전계에서의 방사 적분 공식을 사용하여 전계를 구할 수 있다.

$$\vec{E}(\vec{r}) = \frac{-j\omega\mu}{4\pi}\left(\frac{e^{-jkr}}{r}\right) \cdot \sum_{n=1}^{N}\left[S_n\{\vec{J}(\vec{r}') - \hat{r}(\vec{J}(\vec{r}') \cdot \hat{r})\}e^{jk\vec{r}'\cdot\hat{r}}\right] \quad (6)$$

여기서 $\omega = 2\pi f$, μ는 진공에서의 투자율, S_n은 n번째 삼각형 패치의 면적, $\vec{r}'$은 삼각형 패치의 무게중심의 위치 벡터, $\hat{r}$은 삼각형 패치에서 관측지점으로 향하는 방향벡터, N은 표적을 이루는 삼각형 패치의 총 개수이다.

식 (5)의 Σ항을 각 패치별로 GPU에서 계산하도록 구현함으로써 GPU 기반의 가속이 가능하다. 본 논문에서는 CUDA를 사용하여 구현하였다.

Ⅲ. 시뮬레이션

3-1 정확도 및 계산 속도 검증

본 논문에서 제안한 방법의 계산 결과를 검증하기 위하여 제안한 방법으로 계산한 결과를 상용 소프트웨어인 FEKO의 물리광학법으로 계산한 결과와 비교하였다. 본 논문의 모든 시뮬레이션은 Intel(R) Xeon(R) CPU E5-26700 @ 2.6 GHz Dual core CPU와, 128 GB 메모리, NVIDIA GeForce GTX 650 Ti 그래픽 카드를 장착한 환경에서 수행되었다. 제안된 방법과 FEKO를 이용한 시뮬레이션은 모두 동일한 환경에서 수행되었다. 그림 4는 구에 대하여 주파수를 변화시켜 가며 RCS를 계산한 결과와 계산에 걸린 시간을 보여준다. 그림 4에서 구에 대한 RCS 계산 결과는 FEKO와 제안한 방법으로 개발한 코드 (DARSIM)가 완벽하게 일치한다. 또한 단순히 비교하였을 때, DARSIM이 FEKO에 비하여 256배 이상 빠른 계산 속도를 보여준

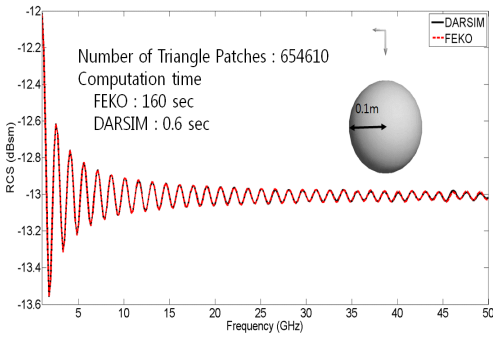


그림 4. 구에 대한 RCS 해석 결과
Fig. 4. RCS analysis results for the sphere(frequency: 1~50 GHz, 256 points, radius: 0.1 m, patch number: 654,610).

다. 그러나 FEKO는 주파수 변경 시 은면 계산을 다시 수행하고, DARSIM은 이전에 계산한 은면 처리 정보를 활용하는 등 동작 방식의 차이가 존재하므로 이러한 비교는 계산 시간에 대한 공정한 비교로 보기 어렵다.

그림 5는 임의의 곡면을 포함한 반경 1.95 m의 날개를 가진 풍력 블레이드에 대하여 블레이드의 회전 방향으로 각도를 변화시켜 가며 RCS를 계산한 결과와 계산에 걸린 시간을 보여준다. 개발된 코드에 전통적인 octree를 제외한 가속 기법을 적용하지 않았을 경우에는 FEKO보다 더

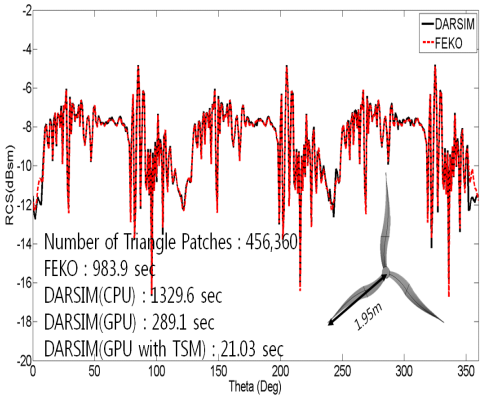


그림 5. 날개가 3개인 블레이드에 대한 RCS 해석 결과
Fig. 5. Results of RCS analysis for blades with three wings(frequency: 10 GHz, theta: 0~360 deg, phi: 0 deg, 361 points, radius: 1.95 m, patch number: 465,360).

긴 계산시간을 필요로 함을 확인할 수 있다. 여기에 GPU 가속을 적용할 경우, CPU로 계산했을 경우에 비하여 21.7 %의 계산시간만을 필요로 하며, 여기에 TSM-RT까지 적용할 경우 CPU로 계산했을 경우에 비하여 1.5 %의 계산시간만을 필요로 함을 확인할 수 있다.

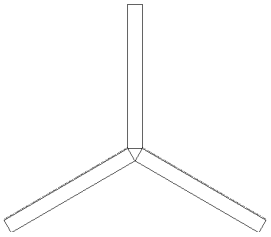
3-2 두 개의 회전 블레이드에 대한 시뮬레이션

본 절에서는 개발한 코드를 사용하여 서로 다른 위치에서 두 개의 대형 블레이드가 함께 회전하는 장면에 대한 동적 RCS를 계산하였다. 표 2는 시뮬레이션에 사용한 파라미터를 보여주며, 표 3은 시뮬레이션에 사용된 표적

표 2. 시뮬레이션 파라미터
Table 2. Simulation parameters.

Parameter	Value
Frequency	23.7～24 GHz
Bandwidth	300 MHz
Number of frequency sample	128
PRF	40 kHz
Observation angle (θ)	90 degree
Observation angle (ϕ)	0 degree

표 3. 시뮬레이션 표적들의 정보
Table 3. Information of the simulation targets.

Parameter	Value
Shape	
Radius	0.5 m
Axis of rotation	z axis
Location of target 1	[−5, 0 0]
Location of target 2	[−10, −10 0]
Rotating speed of target 1	2,000 rpm
Rotating speed of target 2	1,000 rpm
Number of triangle	842,468

의 속성을 보여준다. 그림 6은 계산된 동적 RCS의 시간 거리 그래프를 보여주며, 그림 7은 계산된 동적 RCS의 스펙트로그램을 보여준다.

시뮬레이션에 소요된 시간은 1,304.3초이다. 표 2에서 입사파는 x축 방향에서 입사하므로 두 표적의 거리는 각각 5 m와 10 m에 위치해야 한다. 블레이드는 z축을 축으로 회전하며 한번 회전하는 동안 다가오고 멀어지는 블레이드에 의하여 총 6회의 정반사가 일어난다. 그림 6의 시간 거리 프로파일에서 표적의 거리가 각각 이론값과 일치하는 5 m와 10 m에 위치함을 확인할 수 있으며, 각

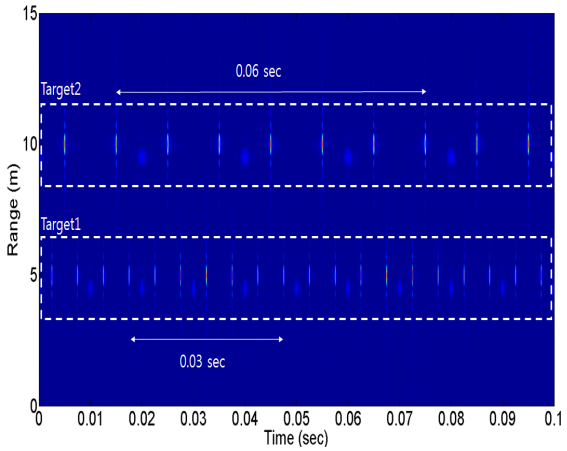


그림 6. 시뮬레이션 결과(시간-거리 그래프)
Fig. 6. Simulation result(time-range graph).

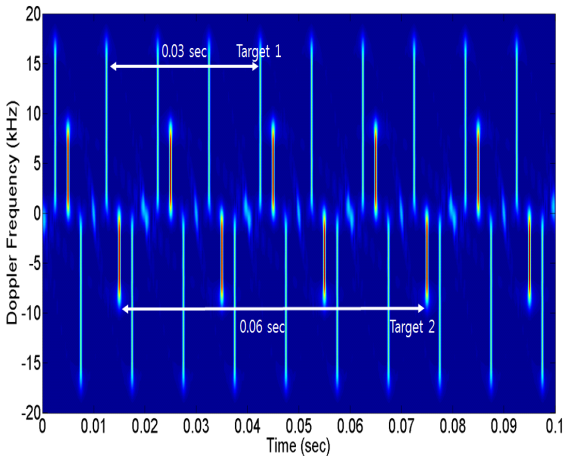


그림 7. 시뮬레이션 결과(spectrogram)
Fig. 7. Simulation result(spectrogram).

표적에서 정반사에 의해 6회의 강한 RCS가 관측되는 간격이 각 표적의 1회전 주기와 일치함을 확인할 수 있다. 그림 7의 스펙트로그램에서는 두 표적의 도플러 성분이 겹쳐진 영상이 나타남을 확인할 수 있다. 스펙트로그램에서는 동일한 3개의 패턴이 하나의 주기를 이루며, 이 주기 역시 그림 7의 결과와 이론값이 일치함을 확인할 수 있다. 또한 두 표적의 블레이드 끝부분에 의해 발생하는 이론적인 도플러 주파수는 식 (7)과 같이 구할 수 있다.

$$f_d = \frac{2v_{tip}}{\lambda} \tag{7}$$

여기서 λ는 레이다의 송신파장이며, 24 GHz에서는 0.0125 m이다. v_{tip}은 블레이드 끝부분의 속력이다. 식 (7)에 표 3에서 설정된 속성을 대입하여 각각의 표적에 대한 이론적인 최대 도플러 주파수를 구하면 16.7 kHz, 8.37 kHz와 같이 얻을 수 있으며, 이는 그림 7에서 관찰되는 최대 도플러 주파수와 일치한다. 이러한 결과는 제안한 방법으로 구현한 해석 소프트웨어가 둘 이상의 서로 다른 회전 표적의 동적 RCS를 올바르게 계산할 수 있음을 보여준다.

Ⅳ. 결 론

본 논문에서는 복잡한 상황에서의 동적 RCS를 계산할 수 있는 시뮬레이터를 제안하고, 이를 구현하기 위한 방법에 대하여 설명하였다. 또한 본 논문에서 제안한 시뮬레이터를 구현하고, 개발한 시뮬레이터의 계산결과를 그 정확성이 검증되어 있는 상용 소프트웨어인 FEKO와 비교하였다. 검증 결과, 개발한 시뮬레이터의 계산 결과는 FEKO의 계산 결과와 동일하였고, 매우 적은 계산 시간을 요구함을 알 수 있었다. 또한 서로 다른 속도로 회전하는 풍력 블레이드의 동적 RCS를 효과적으로 계산할 수 있음을 확인하였다.

본 논문에서 제안한 방법은 계산에 필요한 시간을 단축하기 위하여 해석을 수행하는 표적이 다중반사나 회절의 영향을 매우 적게 받는 단순한 형상이라 가정하였다. 따라서 다중반사나 회절의 영향을 고려해야 하는 표적을 해석할 경우에는 결과값이 부정확해질 수 있다는 한계를 가지고 있다. 본 논문에서 사용하는 표적인 풍력 블레이드는 이러한 영향이 비교적 작아 제안한 방법을 적용하는데 적합하지만, 다른 종류의 표적에 적용할 경우에는 다중반사 및 회절 성분의 영향과 중요성에 대한 검토가 필요하다.

본 논문에서 제안한 기법은 풍력 블레이드의 동적 RCS 계산 이외에도 드론, 헬리콥터 등과 같은 다양한 응용분야에도 적용할 수 있을 것으로 생각된다.

References

[1] Y. F. Lok, A. Palevsky, and J. Wang, "Simulation of radar signal on wind turbine," *IEEE Aerospace and Electronic Systems Magazine*, vol. 26, no. 8, pp. 39-42, Aug. 2011.

[2] B. M. Kent, K. C. Hil, A. Buterbaugh, G. Zelinski, R. Hawley, and L. Cravens, et al., "Dynamic radar cross section and radar Doppler measurements of commercial general electric windmill power turbines Part 1: Predicted and measured radar signatures," *IEEE Antennas and Propagation Magazine*, vol. 50, no. 2, pp. 211-219, Apr. 2008.

[3] B. M. Isom, R. D. Palmer, G. S. Secrest, R. D. Rhoton, D. Saxion, and T. L. Allmon, et al., "Detailed observations of wind turbine clutter with scanning weather radars," *Journal of Atmospheric and Oceanic Technology*, vol. 26, no. 5, pp. 894-910, May 2009.

[4] M. Ritchie, F. Fioranelli, H. Griffiths, and B. Torvik, "Micro-drone RCS analysis," in *2015 IEEE Radar Conference*, Johannesburg, South Africa, 2015, pp. 452-456.

[5] M. Jahangir, C. J. Baker, and G. A. Oswald, "Doppler characteristics of micro-drones with L-band multibeam staring radar," in *2017 IEEE Radar Conference (RadarConf)*, Seattle, WA, 2017, pp. 1052-1057.

[6] A. K. Singh, Y. Kim, "Automatic measurement of blade length and rotation rate of drone using W-band micro-Doppler radar," *IEEE Sensors Journal*, vol. 18, no. 5, pp. 1895-1902, Mar. 2018.

[7] K. B. Kang, I. O. Choi, J. H. Choi, S. G. Sun, J. S. Lee, and B. L. Cho, et al., "Analysis of micro-motion characteristics caused by maneuvering a small unmanned

aerial vehicle acquired from X-band radar," *The Journal of Korean Institute of Electromagnetic Engineering and Science*, vol. 30, no. 7, pp. 573-582, Jul. 2019.

[8] M. K. Bączyk, P. Samczyński, P. Kulpa, and J. Misiurewicz, "Micro-Doppler signatures of helicopters in multistatic passive radars," *IET Radar Sonar & Navigation*, vol. 9, no. 9, pp. 1276-1283, Dec. 2015.

[9] T. Thayaparan, S. Abrol, E. Riseborough, L. J. Stankovic, D. Lamothe, and G. Duff, "Analysis of radar micro-Doppler signatures from experimental helicopter and human data," *IET Radar Sonar & Navigation*, vol. 1, no. 4, pp. 289-299, Aug. 2007.

[10] V. C. Chen, F. Li, S. Ho, and H. Wechsler, "Micro-Doppler effect in radar: Phenomenon, model, and simulation study," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 2-21, Jan. 2006.

[11] S. H. Seol, I. S. Choi, "Classification of warhead and Debris using CFAR and convolutional neural networks," *The Journal of Korean Institute of Information Technology*, vol. 17, no. 6, pp. 85-94, Jun. 2019.

[12] Y. Q. Zhuang, C. X. Zhang, and X. K. Zhang, "A novel simulation approach of aircraft dynamic RCS," *Progress in Electromagnetics Research M*, vol. 36, pp. 85-91, 2014.

[13] T. Milligan, "More applications of Euler rotation angles," *IEEE Antennas and Propagation Magazine*, vol. 41, no. 4, pp. 78-83, Aug. 1999.

[14] NVIDIA, *CUBLAS Library*, Santa Clara, CA, NVIDIA, 2008.

[15] R. Ierusalimschy, *Programming in Lua*, New York, NY, Lua.org, 2006.

[16] F. Weinmann, "Ray tracing with PO/PTD for RCS modeling of large complex objects," *IEEE Transactions on Antennas and Propagation*, vol. 54, no. 6, pp. 1797-1806, Jun. 2006.

[17] L. Guo, T. Fan, "Octree based backward SBR-PO method for electromagnetic scattering of electrically large target," in *2015 IEEE International Conference on Computational Electromagnetics*, *HongKong*, 2015, pp. 303-305.

[18] S. Popov, J. Günther, H. P. Seidel, and P. Slusallek, "Stackless KD-tree traversal for high performance GPU ray tracing," in *Computer Graphics Forum*, Oxford, UK, Sep. 2007, vol. 26, no. 3, pp. 415-424.

[19] A. Breglia, A. Capozzoli, C. Curcio, A. Liseno, and J. Piccinotti, "GPU implementation of hybrid GO/PO BVH-based algorithm for RCS predictions," in *2015 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting*, Vancouver, BC, 2015, pp. 1500-1501.

[20] X. Meng, L. Guo, Y. Wei, and J. Sun, "An accelerated ray tracing method based on the TSM for the RCS prediction of 3-D large-scale dielectric sea surface," *IEEE Antennas and Wireless Propagation Letters*, vol. 14, pp. 233-236, 2015.

최 영 재 [한남대학교/박사과정]

https://orcid.org/0000-0002-8632-5265

2013년 2월: 한남대학교 전자공학과 (공학사)

2018년 2월: 한남대학교 전자공학과 (공학석사)

2018년 3월～현재: 한남대학교 전기전자공학과 박사과정

[주 관심분야] RADAR 신호처리, 특성벡터 추출, 전자파 수치해석

최 인 식[한남대학교/교수]

https://orcid.org/0000-0002-8210-0843

1998년 2월: 경북대학교 전자공학과 (공학사)

2000년 2월: 포항공과대학교 전자전기공학과 (공학석사)

2003년 2월: 포항공과대학교 전자전기공학과 (공학박사)

2003년～2004: LG전자 선임연구원

2004년～2007년: 국방과학연구소 선임연구원

2007년～현재: 한남대학교 전기전자공학과 교수

[주 관심분야] RADAR 신호처리, RADAR 시스템 설계, RCS 해석 및 분석