JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 30 No. 12, pp. 167-175, December 2025
https://doi.org/10.9708/jksci.2025.30.12.167

Efficient Ambient Occlusion Computation in Point Clouds

Jong-Hyun Kim*

*Associate Professor, College of Software and Convergence (Dept. of Design Technology), Inha University, Incheon, Korea

[Abstract]

Ambient Occlusion (AO) is a core technique that enhances depth perception and shape recognition by darkening regions where ambient light is less likely to reach, such as creases and cavities. However, screen-space ambient occlusion (SSAO) assumes mesh connectivity, making it difficult to apply directly to point-set data lacking explicit surface structure. This study proposes an efficient AO computation framework for point sets by combining normal estimation, spherical neighborhood search, planar position evaluation, and normal–directional angle weighting, with min–max normalization and an adjustable intensity parameter to control shading. The system is implemented in Unity3D, representing points as small spherical objects and evaluating local geometric relationships through dot-product and angular calculations to assign AO intensity. Experiments on various point models demonstrate consistent and stable shading effects under different intensity levels. In summary, the proposed method enables accurate and coherent AO visualization without mesh connectivity, offering practical benefits for real-time visualization, scanned-data rendering, and game/VR applications.

▸ **Key words**: Point Set, Ambient Occlusion, Normal Vector Estimation, Real-Time Visualization, Unity3D Framework

[요 약]

주변폐색(AO, Ambient Occlusion)은 오목부·틈새 등 환경광이 적게 도달하는 영역을 어둡게 표현해 깊이감과 형태 인지를 높이는 핵심 기법이다. 하지만 화면공간 기반 SSAO(Screen-Space Ambient Occlusion)는 메시의 연결 정보를 전제로 하므로, 표면·연결성이 없는 포인트셋에는 직접 적용이 어렵다. 본 연구는 포인트셋을 대상으로 법선 추정–구형 이웃 탐색–평면 위치 판정–법선·방향각 가중을 결합하고, min–max 정규화와 강도 파라미터로 음영을 제어하는 효율적 AO 계산 프레임워크를 제안한다. 구현은 Unity3D 기반으로 포인트를 구 오브젝트로 표현해 포인트 환경을 구성하고, 각 포인트의 주변 기하 관계를 내적과 각도 계산으로 평가해 AO 강도를 부여한다. 다양한 모델에 적용한 결과, 강도 조절에 따른 효과 변화와 안정적인 음영 표현을 확인하였다. 요컨대, 제안 기법은 연결 정보 없이도 포인트셋에서 정확하고 일관된 AO 시각화를 가능하게 하여 실시간 시각화, 스캐닝 데이터 렌더링, 게임/VR 응용에 유용함을 보인다.

▸ **주제어**: 포인트셋, 주변폐색, 법선 벡터 추정, 실시간 시각화, Unity3D 프레임워크

• First Author: Jong-Hyun Kim, Corresponding Author: Jong-Hyun Kim
*Jong-Hyun Kim (jonghyunkim@inha.ac.kr), College of Software and Convergence (Dept. of Design Technology), Inha University
• Received: 2025. 10. 10, Revised: 2025. 11. 21, Accepted: 2025. 12. 03.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

3차원 데이터 시각화에서 주변폐색은 물체의 미세한 음영 변화를 표현하여 입체감과 구조적 인식을 높이는 핵심 기술로 널리 활용되고 있다[1,2]. AO는 물체 표면에서 주변 기하가 차단하는 환경광의 양을 계산하여, 오목부나 틈, 인접 면 사이에 상대적으로 어두운 음영을 부여함으로써 깊이감을 강화한다. 이러한 특성으로 인해 AO는 모델링, 게임 그래픽스, 과학 시각화, 건축 렌더링 등 다양한 분야에서 기본적인 조명 기법으로 자리 잡았다.

기존 AO 기법 중 SSAO는 화면 공간에서 깊이 버퍼와 법선 정보를 이용해 AO 효과를 근사적으로 계산하는 방식으로, 계산 효율이 높고 GPU 가속화에 적합하다는 장점을 가진다[3]. 그러나 SSAO는 기본적으로 메시 구조의 표면 연결성을 가정하므로, 표면 정보가 존재하지 않는 포인트셋 데이터에는 직접 적용이 어렵다. 포인트셋은 레이저 스캐닝, 구조광 등으로부터 획득되는 비정형 샘플 데이터로, 메시 변환 없이도 빠른 시각화가 가능하다는 장점이 있지만, 인접 관계나 표면 방향 정보가 명시적으로 존재하지 않아 조명 계산이 어려운 한계를 가진다.

이러한 문제를 해결하기 위해 포인트 클라우드에 AO를 적용하기 위한 다양한 시도가 이루어졌다. 일부 연구에서는 포인트 밀도 기반 근사 조명 모델을 사용하거나, 법선 벡터를 추정하여 국소적 음영 효과를 계산하는 방법을 제안하였다[4,5]. 그러나 이들 방식은 계산 비용이 높거나, 포인트 분포가 불균일한 영역에서 안정적인 음영을 유지하기 어렵다는 한계를 가진다. 특히 포인트 간 거리 변화가 큰 영역에서는 AO 강도가 불안정하게 변하여 시각적 왜곡이 발생하는 문제가 보고되었다.

본 연구에서는 이러한 한계를 보완하기 위해 법선 추정, 구형 이웃 탐색, 평면 위치 판정, 법선·방향각 기반 가중치 계산을 결합한 효율적인 포인트셋 AO 계산 프레임워크를 제안한다. 제안 방식은 각 포인트에 대해 주변 포인트의 공간 분포를 구의 범위 내에서 탐색하고, 평면 방정식을 통해 상대 위치를 판정하며, 내적을 기반으로 AO 강도를 계산한다. 이후 min-max 정규화와 강도 파라미터를 이용하여 다양한 장면에서도 일관된 음영 분포를 유지하도록 설계하였다.

제안 기법은 Unity3D 환경에서 구현되었으며, 포인트를 구 오브젝트로 시각화하여 각 단계별 AO 계산 과정을 검증하였다. 다양한 포인트 모델(예: 구형, 복합 구조 모델 등)에 적용한 결과, 강도 조절에 따라 AO 효과가 안정적으로 변화하며, 기존 SSAO 기반 근사보다 일관된 명암 결과를 제공함을 확인하였다. 본 연구의 접근법은 메시 정보가 없는 포인트셋에서도 정확하고 효율적인 AO 계산을 가능하게 하여, 실시간 시각화, 스캐닝 데이터 렌더링, 게임 및 VR 응용 등 다양한 분야에 유용하게 활용될 수 있다.

II. Related Work

주변폐색은 전역조명의 근사적 형태로, 물체 주변의 환경광 차단 정도를 계산하여 깊이감과 형태 인식을 향상시키는 핵심 기술이다. AO의 개념은 Zhukov 등[2]에 의해 제안되고, Pharr와 Green[1]에 의해 GPU 기반으로 정식화되면서 효율적 실시간 조명 기법으로 발전하였다. 이후 Landis[8]는 영화용 AO 렌더링을 도입하여 전역조명 보조 요소로 AO를 산업 표준화시켰으며, Cook et al.[4]은 BRDF 기반의 재질 렌더링과 AO를 결합하여 물리적 사실성을 확장하였다.

화면 공간에서 깊이 버퍼와 법선 정보를 이용해 음영을 근사하는 SSAO기법은 Mittring[7]과 Bavoil & Sainz[3]의 연구를 통해 확립되었다. Shanmugam과 Arikan[6]은 GPU 병렬화를 적용한 Horizon-Based AO (HBAO) 를 제안하여 시각 품질을 개선하였고, Ritschel 등[5]은 AO 관련 기법들을 종합적으로 정리하면서 AO가 렌더링 파이프라인에서 차지하는 기술적 위치를 체계화하였다. McGuire 등[9]은 Scalable Ambient Obscurance (SAO) 를 통해 고해상도 환경에서의 효율적 근사를 달성하였으며, Schied 등[10]은 Ground-Truth AO (GTAO) 를 제안하여 정확도와 성능의 균형을 달성하였다.

한편, 메시의 연결 정보를 전제로 하는 SSAO는 포인트셋과 같은 비정형 데이터에는 직접 적용이 어렵다. 이러한 한계를 극복하기 위해 포인트셋 기반 AO 기법이 연구되었다. Botsch와 Kobbelt[11]은 GPU 기반 포인트 렌더링 체계를 정립하였고, Kim et al.[4]은 Point-based AO를 제안하여 불균일한 점 분포에서도 안정적인 음영을 계산하였다. Huang 등[12]은 점 밀도에 따라 AO 강도를 적응적으로 조절하는 Density-Adaptive AO를 제안하였고, Han 등[13]은 법선 벡터 기반의 AO 계산을 통해 포인트셋의 접촉 그림자 표현을 개선하였다. Solomon 등[14]은 스펙트럴 분석을 이용해 AO 근사를 이론적으로 확장하였다.

최근에는 AO 계산의 정확도와 효율을 동시에 확보하기 위한 딥러닝 및 신경 기반 AO(Neural AO) 연구가 활발하다. Zhang 등[15]은 합성곱 신경망(CNN)을 이용해 AO를 실시간 예측하는 Deep AO를 제안하였고, Kim 등[16]은

신경 필터링(Neural AO Filtering)을 통해 노이즈 제거와 빠른 재구성을 달성하였다. 나아가 Mildenhall 등[17]의 Neural Radiance Fields (NeRF) 연구는 AO 및 그림자 정보를 보조 입력으로 사용하여 고정밀 장면 복원을 실현하였다.

AO 계산의 실시간성 향상을 위한 GPU 및 엔진 통합 연구도 병행되고 있다. Schütz 등[18]은 GPU 래스터화 기반 AO 계산을 통해 대규모 포인트 클라우드에서도 대화형 렌더링 성능을 확보하였고, Wald 등[19]은 수억 개 포인트 데이터에 대한 AO를 실시간으로 처리할 수 있는 병렬 가속화를 제시하였다. 또한 Bauszat 등[20]은 가이드 이미지 필터링을 활용해 AO 결과의 품질을 유지하면서 후처리 비용을 줄이는 방법을 제안하였다.

요약하면, 기존 연구들은 AO의 정확도와 효율성을 개선하기 위해 SSAO → HBAO → SAO → GTAO로 발전하였으며, 최근에는 포인트 기반 확장과 신경망 기반 근사로까지 확장되고 있다. 그러나 포인트셋 데이터의 불균일성과 법선 추정의 불안정성으로 인해 여전히 정확한 AO 계산에는 한계가 존재한다. 이에 본 연구는 법선 추정-구형 이웃 탐색-평면 위치 판정-각도 기반 가중치 계산을 결합하여, 연결 정보가 없는 포인트셋에서도 일관되고 효율적인 AO 시각화를 달성하는 프레임워크를 제안한다.

III. The Proposed Scheme

1. Algorithm Overview

본 연구에서는 포인트셋에서 AO효과를 효율적으로 계산하기 위한 절차적 프레임워크를 제안한다. 제안된 알고리즘은 각 포인트의 위치, 깊이, 법선 벡터를 기반으로 주변 기하와의 공간적 관계를 평가하여 음영 강도를 산출한다. 이러한 접근은 메시 기반 AO가 가지는 고비용 사전 계산 문제를 완화하며, 표면 연결 정보가 존재하지 않는 점 데이터에서도 일관된 음영 표현을 가능하게 한다.

기본 아이디어는 기준 포인트 P를 중심으로 반구 형태의 탐색 영역을 설정하고, 그 내부에 위치한 인접 포인트 np를 찾은 뒤, 두 포인트 간의 방향 벡터와 법선 벡터 간의 내적을 통해 조명 감쇠를 계산하는 것이다 (Fig. 1 참조). 내적 값이 작을수록 두 벡터의 각도는 커지며, 이는 상대적으로 어두운 영역(광 차단 정도가 높은 영역)을 의미한다. 반대로 내적 값이 클수록 평탄하거나 노출된 영역임을 나타낸다.

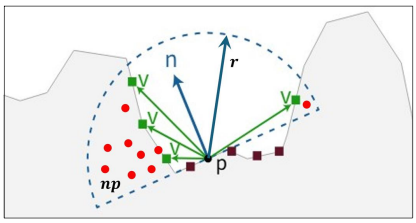

Fig. 1. Point-based AO overview: for each point P, sample a hemisphere of radius r; neighbors np weight shading by the angle between direction and normal.

이 과정은 다음의 단계로 구성된다.

1. 각 포인트의 법선 벡터를 추정한다.
   - 주변 포인트의 공간 분포를 기반으로 가중 평균을 수행하고, 정규화 과정을 통해 법선 방향을 안정화한다.
2. 기준 포인트 P에 대해 지정된 반경 r 내의 이웃 포인트 np를 탐색한다.
   - 구의 방정식 를 이용해 구 내부 여부를 판단한다.
3. 인접 포인트가 기준 포인트의 법선 평면 위에 존재하는지를 계산한다.
   - 평면의 방정식에 인접 포인트의 좌표를 대입하여, 그 결과가 0에 가까울수록 동일 평면에 위치한다고 판단한다.
4. 각 인접 포인트에 대해 방향 벡터와 법선 벡터의 내적을 계산하여 각도를 구한다.
   - 각도값은 AO 강도를 결정하는 핵심 인자로, 최소-최대 정규화를 통해 0-1 범위로 정규화된다.
5. 정규화된 각도값을 색상 스케일로 변환하여 음영 강도를 시각적으로 표현한다.
   - 강도 조절 파라미터 Value를 통해 전체 AO 세기를 제어할 수 있다.

이 과정을 통해, 포인트셋 데이터는 메시 구조 없이도 인접 포인트 간의 기하적 차폐 관계를 효율적으로 계산할 수 있다. 각 포인트의 음영은 반구 내 각도 분포에 따라 부드럽게 변화하며, 결과적으로 깊이감이 강조된 시각화를 얻을 수 있다.

2. Ambient Occlusion in Points

본 연구는 포인트셋 환경에서 AO를 계산하기 위해, Unity3D 상에서 각 포인트를 구 오브젝트로 표현하고 메시 없이도 이웃 관계를 판정할 수 있는 절차를 구성한다. 핵심은 기준 포인트 P의 법선 n을 기준으로 한 반구영역을 설정하고, 그 내부 이웃 np가 만들어내는 방향 벡터 $d = \overrightarrow{Pnp}$와 법선 n 사이의 각도를 기반으로 AO 가중치

를 평가하는 것이다.

먼저 기준 포인트의 법선을 설정(또는 추정)하고, 해당 법선 방향을 따라 반구 표본 공간을 시각화한다. 이때 무작위 방향 벡터 집합을 생성해 n과의 내적 부호(±)로 반구 포함 여부를 판정하여, 법선 기준 반구를 직관적으로 확인한다. 이후 P 주변에 임의(또는 데이터 기반)의 포인트들을 배치하고, (i) 구 반경 r 내부 여부와 (ii) 법선 평면 상·하 위치를 통해 유효 이웃을 선별한다. 유효 이웃에 대해 $\angle(d,n)$을 계산하고, 이 각도(또는 $\cos\theta$)를 AO 기여도로 사용한다. 시각적 확인을 위해 이웃을 범주화(예: 구 내부&평면 위, 최소 각도 후보, 범위 외 등)하여 색상으로 구분한다. 마지막으로, 가장 작은 각도를 갖는 이웃 소수(예: 상위 3개)를 연결하면, 기준 포인트 주변의 차폐 방향성이 직관적으로 드러나며 접촉 음영의 방향적 경향을 파악할 수 있다. 이 절차는 메시 연결이 없어도 거리-평면-각도의 단순한 기하 판정만으로 포인트셋의 AO를 안정적으로 근사할 수 있게 하며, 이후 절(법선 추정, 구 범위 이웃 탐색, 평면 판정, 각도·정규화·강도 제어)에서 세부 수식을 통해 구현을 완결한다.

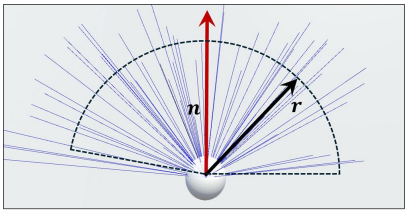

Fig. 2. Normal-hemisphere setup around point P; random directions are classified by the sign of $n \cdot d$.

Fig. 2는 기준 포인트 P의 법선벡터 n을 기준으로 반구 형태의 탐색 영역을 설정하는 과정을 보여준다. 무작위로 생성된 방향 벡터들은 $n \cdot d$의 부호에 따라 반구 내부 또는 외부로 분류되며, 이를 통해 AO 계산에 사용할 유효한 이웃 포인트 후보를 시각적으로 확인할 수 있다.

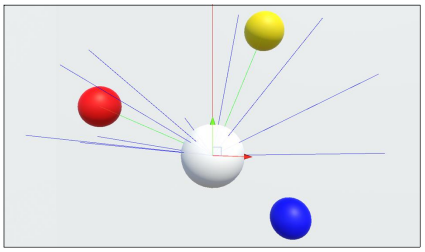

Fig. 3. Neighbor selection: points inside radius r and on the normal plane are highlighted for AO contribution.

Fig. 3은 기준 포인트 P 주변에서 구 반경 r과 법선 평면을 기준으로 유효 이웃을 선별하고, 그 관계를 색으로 표시한 예를 보여준다. 먼저 인접 오브젝트와 P 사이의 거리를 계산해 r과 비교하여 구 내부 여부를 판정하고, 내부로 판정된 경우 Unity 상의 좌표로 법선 방향 평면 위에 있는지 추가로 확인한다. 두 조건(구 내부 ∧ 평면 위)을 모두 만족하는 이웃은 빨간색으로 표시되며, 이들 중 법선 n과 방향벡터 $d = Pn$의 내적(또는 각도)이 가장 작은 이웃은 노란색으로 강조된다. 한편, 구 범위 밖에 있는 오브젝트는 파란색으로 표현하여 AO 계산에 기여하지 않는 대상을 명확히 구분한다.

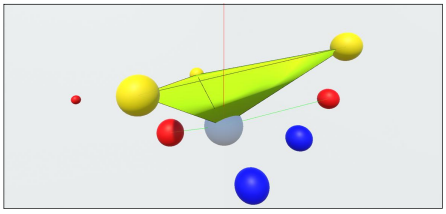

Fig. 4. Connecting the three neighbors with the smallest $\angle(d,n)$ reveals directional occlusion around P.

그림 4는 구 방정식과 평면 방정식을 통해 유효 이웃(구 내부 ∧ 법선 평면 위)으로 판정된 포인트들 가운데, 법선 n과 방향벡터 $d = Pn$의 내적으로 계산한 각도를 기준으로 상위 기여 후보를 선별하는 과정을 보여준다. 기준 포인트 P에서 각 유효 이웃까지의 방향벡터를 구한 뒤, $\angle(d,n)$을 계산하여 각도-위치 리스트에 저장하고, 이를 정렬해 최소 각도를 갖는 이웃들을 찾는다. 마지막으로 가장 작은 각도를 갖는 세 개의 이웃을 P와 연결하면, P 주변에서 차폐가 가장 강하게 발생하는 방향성이 삼각뿔 형태로 시각화되어 AO 기여의 공간적 구조를 직관적으로 확인할 수 있다.

2.1. Calculating Normal Vector in Points

포인트 데이터에 AO를 적용하려면 각 포인트의 위치, 깊이, 법선 벡터가 필요하다. 위치 정보는 외부 파일에서 입력받고, 각 포인트의 법선을 얻기 위해 현재 기준 포인트 주변의 $3\times3\times3$범위에서 이웃 포인트를 탐색한다. 이후 이웃과의 거리와 방향을 계산하여 가중치를 부여해 벡터를 누적하고, 마지막으로 정규화를 수행하면 각 포인트의 법선 벡터를 안정적으로 추정할 수 있다.

2.2. Calculating Neighbor Points in Range

AO 기여도를 평가하기 위해 기준 포인트 P 주변의 유효 이웃을 먼저 선별한다. 본 연구에서는 (1) 구 범위 내부 여부와 (2) 법선 평면 상·하 위치를 순차적으로 판정한다.

Algorithm 1 In Sphere

Input: Position vector of reference object $P = (P_x, P_y, P_z)$
  Position vector of adjacent object $nP = (nP_x, nP_y, nP_z)$
Output: Euclidean distance squared between P and nP

$dx \leftarrow nP_x - P_x$
$dy \leftarrow nP_y - P_y$
$dz \leftarrow nP_z - P_z$
EuclideanDistanceSquared $\leftarrow dx^2 + dy^2 + dz^2$
return EuclideanDistanceSquared

Fig. 5. Pseudocode for determining whether neighboring points lie within the spherical range of radius r.

Algorithm 2 Calculate Plane Equation

Input: normal (the normal vector of the reference particle), P (position of the reference particle), nP (position of the adjacent particle)
Output: EQPres (calculated plane equation result) and classification (plane classification result)

// The normal vector of the plane is the same as the normal vector of P
// The point on the plane is the position of P
// Calculate the constant D using the point P
$D \leftarrow -(normal[0] \times P[0] + normal[1] \times P[1] + normal[2] \times P[2])$

// Substitute the point nP into the plane equation
$EQPres \leftarrow normal[0] \times nP[0] + normal[1] \times nP[1] + normal[2] \times nP[2] + D$

// Classify based on the value of EQPres
if $0 < EQPres \leq 1$ **then**
  classification $\leftarrow$ "On the plane"
else if $EQPres \approx 0$ **then**
  classification $\leftarrow$ "Perpendicular to the plane"
else
  classification $\leftarrow$ "Below the plane"
end if

return $EQPres$, classification

Fig. 6. Pseudocode for checking if neighboring points are located on or below the reference plane using the plane equation $Ax + By + Cz + D = 0$.

1) 구 내부 판정(거리 기반): 인접 포인트 np와 P사이의 유클리드 거리 $\| P - np \|$를 계산하고, 이를 미리 정한 반경 r과 비교한다. $\| P - np \| < r$이면 np를 구 내부 이웃으로 간주한다. 이 절차는 Fig. 5의 의사코드(3번째 줄:거리 계산 및 반경 비교)에 해당한다.
2) 법선 평면 판정(위치 기반): 구 내부로 판정된 이웃에 한해, P의 법선 벡터 $n = (A, B, C)$과 P의 위치를 이용해 평면 상수 D를 계산한다(예: $D = -(Ax_P + Bx_P + Cz_P)$). 이후 이웃 좌표 (x, y, z)를 평면식 $Ax + By + Cz + D$에 대입하여 부호를 확인한다. 결과가 0에 가까울수록 평면에 가깝고, 양수이면 평면 위, 음수이면 평면 아래에 위치한다. 이 로직은 Fig. 6의 의사코드(7번째 줄: D계산, 10번째 줄: 대입·부호 확인, 12-21번째 줄: 분기처리)에 대응한다.
3) 유효 이웃 확정 및 표시(시각화): 두 조건을 모두 만족하는 이웃을 유효 이웃으로 확정하고, 시각적으로 강조 표시한다(예: 내부·평면 위=빨간색). 이후 $d = \overrightarrow{Pnp}$와 n사이의 각도(또는 내적 $n \cdot \hat{d}$)를 계산해 AO 가중치에 활용하며, 최소 각도 이웃은 별도의 강조(예: 노란색)한다. 구 범위 밖 이웃은 비기여 이웃으로 구분(예: 파란색)한다.

2.3. Calculating Angles and Expressing Ambient Occlusion Lighting Effects

본 절에서는 기준 포인트 P의 법선 벡터와 인접 포인트 nP의 방향 벡터 사이의 각도를 계산하고, 이를 바탕으로 주변폐색(AO) 강도를 표현하는 과정을 설명한다.

1) 각도 계산 단계 : 먼저 기준 포인트 P와 각 인접 포인트 nP의 위치 정보를 이용해 방향 벡터 $d = \overrightarrow{Pnp}$를 계산한다. 이후 기준 포인트의 법선 벡터 n과 방향 벡터 d를 각각 정규화 하여 내적으로 두 벡터 간의 각도 θ를 구한다. 이 과정은 Fig. 7의 의사코드 4-8번째 줄에 해당하며,
   - 4~5번째 줄: 인접 포인트의 방향 벡터 계산
   - 6번째 줄: 법선 및 방향 벡터 정규화
   - 8번째 줄: 내적을 통해 $\cos\theta$ 계산
   - 9번째 줄: 라디안 값을 각도로 변환으로 구성
   - 모든 인접 포인트에 대해 계산된 각도는 *angleList*라는 리스트에 위치 좌표와 함께 저장된다. 이후 리스트를 오름차순으로 정렬하여 가장 작은 각도(최대 차폐 영역) 와 가장 큰 각도(노출 영역) 를 구분한다.

Algorithm 3 Calculate Angle

Input: p (position of the reference particle), np (position of the adjacent particle), $normal$ (the normal vector), $neighbors$, k (the index of the neighbor), $_particles$, $angleList$
Output: Sorted $angleList$ with angle values and adjacent particle positions

$VecIP \leftarrow [np[0] - p[0], np[1] - p[1], np[2] - p[2]]$ ▷ Compute the vector from P to nP
$dir \leftarrow neighbors[k] - >_p os -_p articles[k] - >_p os$ ▷ Compute the vector from the neighboring particle to the reference particle
$dir.Normalize()$ ▷ Normalize the direction vector

$theta \leftarrow dir \cdot normal$ ▷ Compute the dot product of the direction vector and the normal vector
$theta \leftarrow \arccos(theta) \times \frac{180.0}{\pi}$ ▷ Convert the angle from radians to degrees

$angleList \leftarrow angleList.add(\theta, [np[0], np[1], np[2]])$ ▷ Add the angle and the adjacent particle position to the list

$angleList.sort()$ ▷ Sort the angle list in ascending order

Fig. 7. Pseudocode for calculating the angle between the normal and direction vectors. The algorithm computes the normalized dot product between n and d, stores each angle in a list, and sorts them to identify minimum and maximum values.

2) AO 조명 강도 계산 및 시각화 : 계산된 각도 값들은 그대로 사용할 경우 범위가 불균일하므로, 최소-최대 정규화를 적용하여 0-1 사이의 스케일로 변환한다. 정규화 식은 다음과 같다 : $NormAngle = \frac{x - \min(\theta)}{\max(\theta) - \min(\theta)}$. 여기서 x는 특정 포인트의 각도, $\min(\theta)$와 $\max(\theta)$는 전체 이웃 포인트 중 최소·최대 각도값이다. 정규화된 값은 R, G, B 성분의 최대값(255)을 곱해 음영 강도를 계산하며, Value 파라미터(0-1 사이) 를 통해 전체 명암 대비를 조절할 수 있다. 이 일련의 과정은 Fig. 8의 의사코드에 제시된 바와 같이,

- 4번째 줄: 정규화 각도 계산
- 5번째 줄: Value를 이용한 강도 조정
- 이후: 색상으로 변환 및 AO 강도 적용으로 구성

Algorithm 4 Draw Angle

1: **Input:** *_particles* (list of particles), *allminAngle* (minimum angle), *Value* (constant)
2: **Output:** Normalized values and weight w
3: **for all** p in *_particles* **do**
4: $normalVal \leftarrow \frac{(p\_angle - allminAngle)}{(20.0 - allminAngle)} \cdot 255.0$
5: $normalVal \leftarrow normalVal \cdot Value$
6: $w \leftarrow 1.0 - \frac{normalVal}{255.0}$
7: **end for**

Fig. 8. Pseudocode for applying AO lighting effects and adjusting shading intensity. Normalized angles are scaled by RGB intensity and a global parameter Value to control overall shading contrast.

IV. Experiment and Results

1. Experimental Setup Overview

제안 기법은 Unity3D 상에서 포인트를 구(Sphere) 오브젝트로 표현하여 구현하였고, 각 포인트에 대해 법선-방향각 기반 AO 가중치를 계산한 뒤 min-max 정규화와 Value 파라미터로 음영 강도를 제어하였다. 정규화된 각도값은 RGB 최대값과 곱해 시각적 강도를 결정하며, Value(0-1)를 통해 전체 대비를 조정한다.

본 연구에서 사용된 포인트셋 모델은 크게 기하학적 기본 형상과 스캔 기반 데이터의 두 부류로 구성된다. 기하학적 모델은 구(sphere), 토러스(torus), 간단한 복합 구조(mesh 조합) 등을 삼각 메시로부터 균일 샘플링하여 생성하였으며, 포인트 개수는 모델 종류에 따라 대략 50,000~300,000점 범위로 설정하였다. 이들 데이터는 주로 균일 또는 준균일 분포를 가지도록 샘플링하여, 형상에 따른 AO 패턴 변화를 관찰하는 데 초점을 두었다. 스캔 기반 포인트셋은 내부적으로 구축한 실측 데이터로, 일부 영역에서 점 간 간격이 불균일하게 분포하는 특징을 가지며, 수만~수십만 점 규모로 구성된다.

2. Intensity Parameter Variation

Fig. 9는 Value를 1.0, 0.8, 0.6, 0.5로 변화시키며 동일 모델에서 AO 강도의 변화를 관찰한 결과이다. Value가 작아질수록 음영 강도가 약해지고, Value가 커질수록 더 강한 차폐 효과가 나타남을 확인했다(상단 좌측부터 순서대로 1.0 → 0.8 → 0.6 → 0.5). 이는 제안 프레임워크가 사용자의 의도에 따라 일관되게 조절 가능한 명암 대비를 제공함을 보여준다. 분석 결과는 다음과 같다.

- 단조성: Value 증가에 따른 음영 강화가 단조적으로 관찰되어 파라미터 해석이 직관적이다.
- 정규화 효과: min-max 정규화로 각 장면의 각도 분포가 0-1로 표준화되어, 모델과 장면이 달라져도 유사한 Value가 유사한 시각 효과로 대응된다.

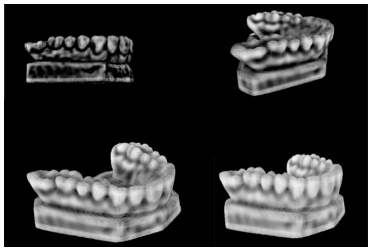

Fig. 9. AO intensity comparison with Value = {1.0, 0.8, 0.6, 0.5} (top-left to bottom-right). Lower Value yields weaker shading; higher Value strengthens occlusion.

3. Application to Various Point Models

Fig. 10은 서로 다른 형상의 포인트 모델에 제안 기법을 적용한 결과로, 오목부·틈·접촉부 등에서 AO가 안정적으로 작동함을 보여준다. 본문 결론에서도 다양한 모델에 대한 실험에서 대부분 안정적인 결과를 보였음을 언급한다. 분석 결과는 다음과 같다.

- 형상 일반화: 국소 차폐 구조가 모델 형태에 관계없이 일관되게 강조되어, 포인트셋 기반 형상 인지에 유리하다.
- 메시 비종속성: 표면/연결 정보 없이도 법선-방향각과 정규화만으로 명암이 안정화되며, 이는 포인트 데이터의 비정형성에 대응하는 핵심 장점이다.

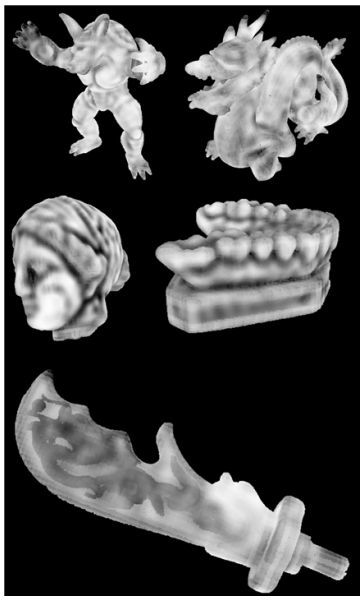

Fig. 10. AO results on diverse point-set models using the proposed method; stable occlusion across different shapes.

4. Visual Quality Observation

- 연속성: 각도 기반 가중치와 정규화의 결합으로 모델 표면에 걸쳐 부드러운 음영 전이가 나타난다.
- 대비 제어성): 동일 장면에서도 Value만 조정해 명암 대비를 미세 튜닝할 수 있어, 시각화 목적(강조/완화)에 맞춘 유연한 톤 매핑이 가능하다.

제안 기법의 효율성을 확인하기 위해 포인트 개수에 따른 CPU 구현과 GPU Compute Shader 구현의 처리 시간을 비교하였다. 포인트 수를 10,000, 50,000, 100,000, 300,000으로 증가시키며 측정한 결과, CPU 측 연산 시간은 각각 약 18 ms, 85 ms, 170 ms, 540 ms 수준으로 급격히 증가한 반면, GPU에서는 동일 조건에서 약 4 ms, 7 ms, 12 ms, 28 ms로 상대적으로 작은 증가폭을 보였다. 이를 기준으로 했을 때 GPU는 포인트 수에 따라 약 4.5배에서 최대 19배까지 빠른 처리 속도를 달성하였으며, 30만 포인트 환경에서도 약 35 FPS 내외의 프레임레이트를 유지하여 대규모 포인트셋에 대해서도 실시간 AO 시각화가 가능함을 확인할 수 있었다.

V. Experimental Environment

본 연구에서 제안한 포인트셋 기반 주변폐색 계산 프레임워크의 성능을 검증하기 위해, Unity3D 엔진 환경에서 일련의 실험을 수행하였다. 실험 환경은 하드웨어 사양, 소프트웨어 구성, 데이터셋 구성으로 구분하여 다음과 같이 설정하였다.

1. Hardware Configuration

실험은 고해상도 포인트셋의 실시간 렌더링 성능을 검증하기 위해 고성능 GPU를 장착한 워크스테이션에서 수행되었다.

- CPU: Intel Core i9-13900K (24 cores, 3.0 GHz)
- GPU: NVIDIA GeForce RTX 4090 (24 GB GDDR6X)
- Memory: 64 GB DDR5
- Display: 3840 × 2160 (4K) resolution
- Operating System: Windows 11 Pro 64-bit

이 구성은 포인트 수가 수십만 개 이상인 데이터셋에서도 AO 계산과 시각화를 실시간으로 처리하기에 충분한 성능을 제공한다.

2. Software Configuration

제안 기법은 Unity 2022.3 LTS 버전에서 구현되었으며, C# 스크립트를 이용해 각 포인트의 위치·법선·AO 강도를 계산하였다. 주요 기능은 GPU 가속을 활용한 Compute Shader 기반으로 작성되어 AO 가중치 계산과 정규화 과정을 실시간으로 수행한다.

- Engine: Unity 2022.3 (LTS, URP pipeline)
- Language: C# / HLSL (Compute Shader)
- Framework: .NET 6.0
- Rendering Mode: Deferred Rendering와 Screen-space Blending

렌더링 단계에서 AO 강도는 RGB 256단계 색상으로 변환되어 음영 강도를 표현하였으며, AO 파라미터(Value = 0.5 – 1.0)와 반경 r 값은 실험별로 조정하였다.

3. Comparison with Existing AO Methods

동일한 포인트셋 장면에 SSAO와 HBAO를 적용하여 연산 시간과 프레임레이트를 비교한 결과, 제안 기법은 별도의 메시 변환 없이도 이들 기법과 유사한 AO 품질을 유지하면서 포인트 수 증가에 대해 더 안정적인 성능 저하 양상을 보였다.

VI. Conclusion

본 연구에서는 메시 구조나 표면 연결 정보가 없는 포인트셋 환경에서 주변폐색(Ambient Occlusion, AO)을 효율적으로 계산하기 위한 프레임워크를 제안하였다. 기존의 SSAO기법은 메시 기반 데이터에 의존하기 때문에 포인트 클라우드에서는 직접 적용이 어려웠다. 이에 본 연구는 법

선 벡터 추정, 구 범위 내 이웃 탐색, 법선 평면 판정, 법선-방향각 기반 가중치 계산, 정규화 및 강도 제어의 절차로 구성된 새로운 접근법을 설계하였다.

제안된 방법은 Unity3D 엔진 환경에서 구현되었으며, 다양한 포인트셋 모델에 적용한 결과 다음과 같은 특징을 확인하였다.

1) 정확성: 각도 기반 AO 계산과 정규화 과정을 통해 불균일한 포인트 분포에서도 안정적인 음영이 형성되었다.
2) 효율성: 메시 정보 없이도 단순한 기하 연산만으로 AO를 근사하여 실시간 시각화에 적합한 성능을 확보하였다.
3) 일관성: Value 파라미터 조절만으로 AO 강도를 쉽게 제어할 수 있어, 다양한 모델과 장면에서도 균일한 시각 품질을 유지하였다.

결과적으로, 본 프레임워크는 포인트셋 기반 데이터의 구조 인식과 시각적 입체감을 효과적으로 향상시키며, 실시간 시각화·스캐닝 데이터 렌더링·게임 및 VR 응용 등 다양한 분야에서 활용 가능성이 있다. 향후 연구에서는 AO 계산을 GPU 병렬 처리와 결합하고, 학습 기반 AO 근사(Neural AO) 기법을 통합하여 대규모 포인트 클라우드의 실시간 조명 표현으로 확장할 계획이다.

ACKNOWLEDGEMENT

The National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2023-00254695, Contribution Rate : 50%). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2022-00155915, Artificial Intelligence Convergence Innovation Human Resources Development (Inha University), Contribution Rate : 50%)

REFERENCES

[1] Matt Pharr and Simon Green, “Ambient Occlusion,” GPU Gems 1, pp. 279-292, Addison-Wesley, 2004.

[2] Sergey Zhukov, Alexei Iones, and Gregory Kronin, “An Ambient Light Illumination Model,” in Rendering Techniques (Eurographics Workshop on Rendering), pp. 45-55, Springer, 1998. DOI: 10.1007/978-3-7091-6453-2_5

[3] Louis Bavoil and Miguel Sainz, “Screen Space Ambient Occlusion,” NVIDIA Developer Information, 2008. Available: https://developer.nvidia.com

[4] Hyun-Kyu Kim, Jae-Young Choi, and Joon-Hyung Park, “Point-based Ambient Occlusion for Rendering Scattered Data,” Computer Graphics Forum, vol. 27, no. 7, pp. 1877-1884, 2008. DOI: 10.1111/j.1467-8659.2008.01339.x

[5] Tobias Ritschel, Thorsten Grosch, and Hans-Peter Seidel, “State of the Art in Ambient Occlusion,” Computer Graphics Forum, vol. 28, no. 4, pp. 1369-1394, 2009. DOI: 10.1111/j.1467-8659.2009.01429.x

[6] P. Shanmugam and S. Arikan, “Hardware Accelerated Ambient Occlusion Techniques on GPUs,” Proceedings of ACM I3D, pp. 73-80, 2007. DOI: 10.1145/1230100.1230114

[7] M. Mittring, “Finding Next Gen: CryEngine 2,” SIGGRAPH Advanced Real-Time Rendering Course, 2007.

[8] H. Landis, “Production-Ready Global Illumination,” SIGGRAPH Course Notes, 2002.

[9] M. McGuire, B. Mara, and D. Nowrouzezahrai, “Scalable Ambient Obscurance,” Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, 2012. DOI: 10.1145/2159616.2159631

[10] C. Schied, M. Salvi, and H. Lensch, “Ground-Truth Ambient Occlusion: Fast AO with Accurate Results,” Proceedings of High Performance Graphics, 2017. DOI: 10.2312/hpg.20171097

[11] M. Botsch and L. Kobbelt, “High-quality Point-based Rendering on Modern GPUs,” Pacific Graphics, 2003.

[12] J. Huang, X. Wang, and Y. Chen, “Density-Adaptive Ambient Occlusion for Point Cloud Visualization,” IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 3, pp. 1418-1430, 2020. DOI: 10.1109/TVCG.2019.2934691

[13] J. Han, S. Lee, and B. Gooch, “Normal-Guided Ambient Occlusion for Point Cloud Rendering,” Computers & Graphics, vol. 88, pp. 76-87, 2020. DOI: 10.1016/j.cag.2020.03.004

[14] J. Solomon, R. Rustamov, and L. Guibas, “Spectral Methods for Point-based AO Approximation,” ACM Transactions on Graphics, vol. 39, no. 4, 2020. DOI: 10.1145/3386569.3392461

[15] Y. Zhang, H. Li, and M. Zhou, “Deep Ambient Occlusion: Learning-based Approximation for Real-time Rendering,” IEEE Access, vol. 8, pp. 129871-129883, 2020. DOI: 10.1109/ACCESS.2020.3009332

[16] J.-H. Kim, M. Kim, and S. Lee, “Neural AO Filtering for Real-time Shading,” Computers & Graphics, vol. 104, pp. 67-78, 2022. DOI: 10.1016/j.cag.2022.02.004

[17] B. Mildenhall, P. Srinivasan, M. Tancik, et al., “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” European Conference on Computer Vision (ECCV),

pp. 405-421, 2020.

[18] M. Schütz, S. Steinberger, and D. Schmalstieg, "Interactive Ambient Occlusion for Large Point Clouds using GPU Rasterization," IEEE VR, pp. 1-10, 2021. DOI: 10.1109/VR50410.2021.00021

[19] I. Wald, A. Dirk, and M. Slusallek, "Real-time Ambient Occlusion Computation for Massive Point Data," IEEE Computer Graphics and Applications, vol. 43, no. 1, pp. 44-55, 2023. DOI: 10.1109/MCG.2023.3235645

[20] P. Bauszat, M. Eisemann, and M. Magnor, "Guided Image Filtering for Interactive Ambient Occlusion," Computer Graphics Forum, vol. 30, no. 2, pp. 209-218, 2011. DOI: 10.1111/j.1467-8659.2011.01853.x

Authors

Jong-Hyun Kim received the B.A. degree in the Department of Digital Contents at Sejong University in 2008. He received M.S. and Ph.D. degrees in the Department of Computer Science and Engineering at Korea University, in 2010 and 2016. Prof. Kim is an Associate Professor in the College of Software and Convergence (Dept. of Artificial Intelligence, Design Technology) in Inha University. His current research interests include fluid animation and virtual reality.